PASS: PATCH-AWARE SELF-SUPERVISION FOR VISION TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent self-supervised representation learning methods have shown impressive results in learning visual representations from unlabeled images. This paper aims to improve their performance further by utilizing the architectural advantages of the underlying neural network, as the current state-of-the-art visual pretext tasks for self-supervised learning do not enjoy the benefit, *i.e.,* they are architecture-agnostic. In particular, we focus on Vision Transformers (ViTs), which have gained much attention recently as a better architectural choice, often outperforming convolutional networks for various visual tasks. The unique characteristic of ViT is that it takes a sequence of disjoint patches from an input image and processes patch-level representations internally. Inspired by this, we design a simple yet effective visual pretext task, coined *Patch-Aware Self-Supervision* (PASS), for learning better patch-level representations. To be specific, we enforce invariance against each patch and its neighbors, *i.e.,* each patch treats similar neighboring patches as positive samples. Consequently, training ViTs with PASS produces more semantically meaningful attention maps patch-wisely in an unsupervised manner, which can be beneficial, in particular, to downstream tasks of a dense prediction type. Despite the simplicity of our scheme, we demonstrate that it can significantly improve the performance of existing self-supervised learning methods for various visual tasks, including object detection and semantic segmentation.

1 INTRODUCTION

Recently, self-supervised learning (SSL) has achieved successful results in learning visual representations from unlabeled images with a variety of elaborate pretext tasks, including contrastive learning (He et al., 2020; Chen et al., 2020a;b), clustering (Caron et al., 2020), and pseudo-labeling (Grill et al., 2020; Caron et al., 2021; Chen & He, 2021). The common nature of their designs is on utilizing different augmentations from the same image as the *positive pairs*, *i.e.*, they learn representations to be invariant to the augmentations. The SSL approaches without utilizing human-annotated labels have been competitive with or even outperformed the standard supervised learning (He et al., 2016) in various downstream tasks, including image classification (Chen et al., 2020b), and object detection (Caron et al., 2021), and image segmentation (Caron et al., 2021).

Meanwhile, motivated by the success of Transformers (Vaswani et al., 2017) in natural language processing (Devlin et al., 2018; Brown et al., 2020), Vision Transformers (ViTs; Dosovitskiy et al. 2020; Touvron et al. 2020; 2021) have gained much attention as an alternative to convolutional neural networks (CNNs) with superior performance over CNNs in various visual tasks (Touvron et al., 2020; Radford et al., 2021). For example, ViT-S/16 (Touvron et al., 2020) has a $1.8\times$ faster throughput than ResNet-152 (He et al., 2016) with a higher accuracy in the ImageNet (Deng et al., 2009) benchmark.

There have been several recent attempts to apply existing self-supervision techniques to ViTs (Chen et al., 2021b; Xie et al., 2021c; Caron et al., 2021). Although the techniques have shown to be also effective with ViTs, they do not fully utilize architectural advantages of ViTs, *i.e.*, their pretext tasks are architecture-agnostic. For example, ViTs are able to process patch-level representations, but pretext tasks used in the existing SSL schemes only use image-level supervision. As a result, existing self-supervised ViTs (Chen et al., 2021b; Caron et al., 2021) tend to produce inaccurate attention maps at the final layer of ViTs for each image patch as shown in the second row of Figure 1.

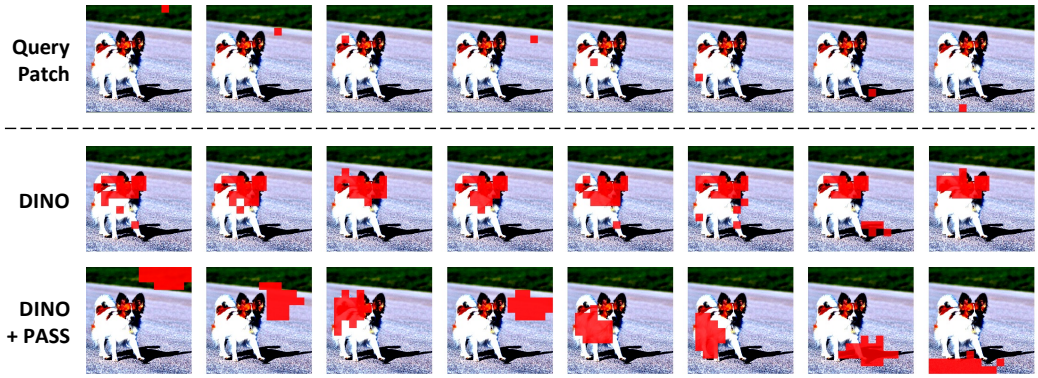

Figure 1: Visualization of top-10% patches obtained by thresholding self-attention maps of query patches (top) in the last layer of ViT-S/16 trained with DINO (middle) and DINO + PASS (bottom). While the selected patches obtained by DINO are not semantically correlated with its query patch, PASS encourages the model to learn semantic correlations among patches.

This limitation inspires us to investigate the following question: *how to utilize architectural characteristics of ViTs for improving the quality of learned representations without human-annotated supervision?*

Contribution. In this paper, we propose *Patch-Aware Self-Supervision (PASS)*, a simple yet effective scheme for learning patch-level representations, which can be beneficial to various visual downstream tasks. Our self-supervised learning module, PASS, can be incorporated into any image-level self-supervision scheme, *e.g.,* DINO (Caron et al., 2021), MoCo-v3 (Chen et al., 2021b) and MoBY (Xie et al., 2021c), for learning both global (*i.e.*, image-level) and local (*i.e.*, patch-level) information simultaneously. Figure 1 shows that PASS enhances the quality of self-attention maps of DINO, which is an evidence that PASS encourages to learn better patch-level representations.

Our key idea for designing PASS is to treat *semantically similar neighboring* patches as positive samples motivated by the following prior knowledge: adjacent patches often share a common semantic context. Since there might be multiple positive patches (but we do not know exactly which patches are positive), we first select a fixed number of candidates for *positive* patches using the cosine similarity between patch representations of the current model. Here, some of them might be noisy (*e.g.*, not positive), and for the purpose of denoising, we utilize an attention-based aggregation module (Touvron et al., 2021) for pruning and summarizing the selected neighboring patches. Then, we minimize the distance between each patch representation and the corresponding summarized one. We provide an overall illustration of PASS in Figure 2.

In our experiments, we incorporate the proposed self-supervised learning scheme, PASS, into the state-of-the-art image-level self-supervision, DINO (Caron et al., 2021).[1] To demonstrate the effectiveness of PASS, we pre-train ViT-S/16 on the ImageNet (Deng et al., 2009) dataset and evaluate the pre-trained ViT on a wide range of dense prediction downstream benchmarks: (a) COCO object detection and instance segmentation (Lin et al., 2014), (b) ADE20K semantic segmentation (Zhou et al., 2017), and (c) DAVIS 2017 video instance segmentation (Pont-Tuset et al., 2017). For example, our method not only improves DINO (Caron et al., 2021) significantly from 31.13 to 38.18 in mIoU metric but also outperforms other baselines, *e.g.,* 37.28 of DetCo (Xie et al., 2021a) and 37.19 of DenseCL (Wang et al., 2021b), simultaneously on the ADE20K semantic segmentation task.

2 Method

In this section, we introduce *Patch-Aware Self-Supervision (PASS)*, a simple yet effective framework for learning better patch-level representations, which is tailored to Vision Transformers (Dosovitskiy et al., 2020) for utilizing their unique architectural advantages. We first review Vision Transformers

[1]Nevertheless, the joint usage of PASS and other image-level SSL frameworks (Chen et al., 2021b; Xie et al., 2021c) is also possible.

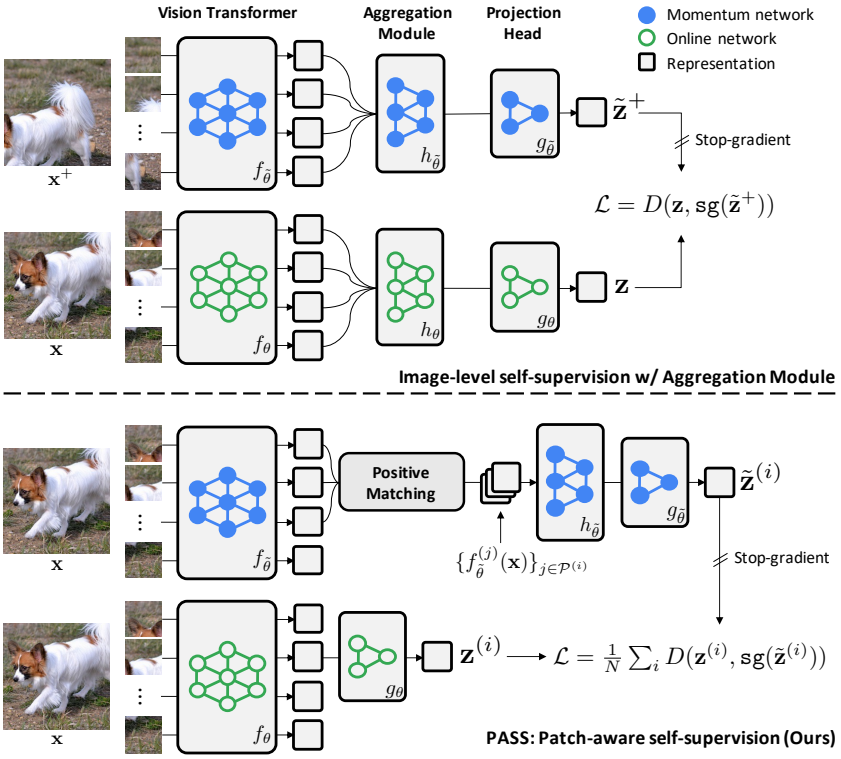

Figure 2: Illustration of the proposed framework, Patch-Aware Self-Supervision (PASS). **Top**: image-level self-supervision, which minimizes the distance between the final representations of two differently augmented images. **Bottom**: patch-aware self-supervision (ours) which minimizes the distance between the final representations of each patch and its positives. We use both types of self-supervision for learning image-level and patch-level representations simultaneously.

with recent self-supervised learning frameworks (Chen et al., 2021b; Caron et al., 2021; Xie et al., 2021c) in Section 2.1, and then present details of PASS in Section 2.2. Figure 2 illustrates the overall framework of our method, PASS.

2.1 Preliminaries: Self-supervised Vision Transformer

Vision Transformers. Let $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ be an image where (H, W) is the resolution of $\mathbf{x}$ and C is the number of channels. Vision Transformers (ViTs; Dosovitskiy et al. 2020) treat the image $\mathbf{x}$ as a sequence of non-overlapping patches $\{\mathbf{x}^{(i)} \in \mathbb{R}^{P^2C}\}_{i=1}^{N}$ (*i.e.*, tokens) where each patch has a fixed resolution (P, P). Then, the patches are linearly transformed to D-dimensional patch embeddings $\mathbf{e}^{(i)} = E\mathbf{x}^{(i)} + E_{\text{pos}}^{(i)} \in \mathbb{R}^D$ where $E \in \mathbb{R}^{D \times P^2C}$ is a linear projection and $E_{\text{pos}}^{(i)} \in \mathbb{R}^D$ is a positional embedding for the patch index i. ViTs also prepend the [CLS] token, which represents the entire patches (*i.e.,* the given image $\mathbf{x}$), to the patch sequence with a learnable embedding $\mathbf{e}^{[\texttt{CLS}]} \in \mathbb{R}^D$. The resulting input sequence $\mathbf{e}$ is $\mathbf{e} = [\mathbf{e}^{[\texttt{CLS}]}; \mathbf{e}^{(1)}; \mathbf{e}^{(2)}; \ldots; \mathbf{e}^{(N)}]$. Then, ViTs take the input $\mathbf{e}$ and output all the patch-level and image-level (*i.e.*, [CLS] token) representations with a transformer encoder.[2] For conciseness, we use f_θ to denote the whole process of a ViT parameterized by θ[3] as follows:

$$f_\theta(\mathbf{x}) = f_\theta([\mathbf{e}^{[\texttt{CLS}]}; \mathbf{e}^{(1)}; \mathbf{e}^{(2)}; \ldots; \mathbf{e}^{(N)}]) = [f_\theta^{[\texttt{CLS}]}(\mathbf{x}); f_\theta^{(1)}(\mathbf{x}); f_\theta^{(2)}(\mathbf{x}); \ldots; f_\theta^{(N)}(\mathbf{x})], \quad (1)$$

[2]We omit the details of the transformer encoder (Vaswani et al., 2017) of which each layer consists of the self-attention module, skip connection and multi-layer perceptron (MLP).

[3]Note that θ contains all the transformer encoder paraemeters and embedding parameters E, E_{pos}, and $\mathbf{e}^{[\texttt{CLS}]}$.

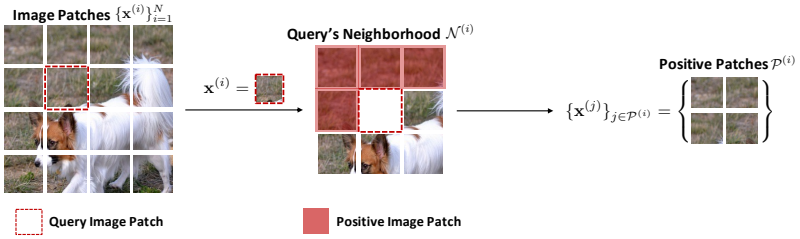

Figure 3: Illustration of positive matching. For a given query patch, we find semantically similar (*i.e.*, positive) patches from its neighborhood using the cosine similarity on the representation space.

where $f_\theta^{[\texttt{CLS}]}(\mathbf{x})$ and $f_\theta^{(i)}(\mathbf{x})$ are the final representations of the $[\texttt{CLS}]$ token and the i-th patch, respectively. Remark that $f_\theta^{[\texttt{CLS}]}(\mathbf{x})$ is utilized for solving image-level downstream tasks such as image classification (Dosovitskiy et al., 2020; Touvron et al., 2020; 2021) while the patch-level representations $\{f_\theta^{(i)}(\mathbf{x})\}_{i=1}^N$ are done for dense prediction tasks, *e.g.,* object detection (Carion et al., 2020) and semantic segmentation (Xie et al., 2021b).

Self-supervised learning with ViTs. The recent literature (Chen et al., 2021b; Xie et al., 2021c; Caron et al., 2021) has attempted to apply self-supervised learning (SSL) frameworks to ViTs. They commonly construct a positive pair $(\mathbf{x}, \mathbf{x}^+)$ by applying different augmentations to the same image, and then learn their representations to be similar, *i.e.*, invariant to augmentations. We here provide a generic formulation of this idea.[4] To this end, we denote two ViT backbone networks as f_θ and $f_{\tilde{\theta}}$, and their projection heads as g_θ and $g_{\tilde{\theta}}$ are parametrized by θ and $\tilde{\theta}$, respectively. Then, the generic form can be written as follows:

$$\mathcal{L}^{\text{SSL}}(\{\mathbf{x}, \mathbf{x}^+\}; \theta, \tilde{\theta}) := D(g_\theta(f_\theta^{[\texttt{CLS}]}(\mathbf{x})), \texttt{sg}(g_{\tilde{\theta}}(f_{\tilde{\theta}}^{[\texttt{CLS}]}(\mathbf{x}^+)))), \tag{2}$$

where D is a distance function and $\texttt{sg}$ is the stop-gradient operation. Note that the distance D and the architecture choice of g depend on the type of self-supervision; for example, Caron et al. (2021) update $\tilde{\theta}$ by the exponential moving average of θ, and use Kullback-Leibler (KL) divergence as D, where the projection heads $g_{\tilde{\theta}}$ and g_θ are designed to produce a probability distribution over the final feature dimension.

We remark that the idea of constructing a positive pair $(\mathbf{x}, \mathbf{x}^+)$ of augmented *images* is architecture-agnostic, which means it does not fully utilize architectural benefits of ViTs. For example, ViTs can handle patch-level representations $\{f_\theta^{(i)}(\mathbf{x})\}$, but the recent SSL frameworks use only $f_\theta^{[\texttt{CLS}]}(\mathbf{x})$ as described in (2). This motivates us to explore the following question: *how to construct patch-aware self-supervision, i.e., positive pairs of patches?*

2.2 PASS: Patch-aware self-supervision

Recall that our goal is to learn better patch-level representations, which can be beneficial to various type of downstream tasks. Our key idea is to consider neighboring patches as positive samples based on the continuous nature of image patches. Overall, PASS aims to learn patch-level representations via predicting self-supervision constructed by the following procedure: for each patch $\mathbf{x}^{(i)}$, *Positive Matching* first finds a set of candidates for positive patch indices $\mathcal{P}^{(i)}$ from its *neighborhood* $\mathcal{N}^{(i)}$ (see Figure 3), and then *Aggregation Module* aggregates their representations $\{f_\theta^{(j)}(\mathbf{x})\}_{j\in\mathcal{P}^{(i)}}$ as self-supervision for $\mathbf{x}^{(i)}$ (see Figure 2).

Neighboring patches. Given a query patch $\mathbf{x}^{(i)}$, we assume that there exists a semantically similar patch $\mathbf{x}^{(j)}$ in its neighborhood $\mathcal{N}^{(i)}$, because neighboring patches $\{\mathbf{x}^{(j)}\}_{j\in\mathcal{N}^{(i)}}$ often share a semantic context with the query $\mathbf{x}^{(i)}$. Let $\{\mathbf{x}^{(j)}\}_{j\in\mathcal{N}^{(i)}}$ be a set of neighboring patches, where $\mathcal{N}^{(i)}$ is a set of patch indices in the neighborhood. We simply use $\mathcal{N}^{(i)}$ to be adjacent patches (*i.e.,* $|\mathcal{N}^{(i)}| = 8$), and empirically found that this choice is important for selecting candidates for positive patches (see Section 4.4 for ablation experiments).

[4]Built upon this formulation, one can additionally consider negative pairs for contrastive learning or asymmetric architectures such as a prediction head (Chen et al., 2021b; Xie et al., 2021c).

Matching positives from the neighborhood. To sample positive (*i.e.*, semantically similar) patches from the neighborhood $\mathcal{N}^{(i)}$, we measure the semantic closeness between the query patch $\mathbf{x}^{(i)}$ and its neighboring patch $\mathbf{x}_\theta^{(j)}$ for all $j \in \mathcal{N}^{(i)}$. To this end, we use the cosine similarity on the representation space, *i.e.*,

$$s(i,j) = f_\theta^{(i)}(\mathbf{x})^\top f_\theta^{(j)}(\mathbf{x}) / ||f_\theta^{(i)}(\mathbf{x})||_2 ||f_\theta^{(j)}(\mathbf{x})||_2.$$

We take *top-k* positive patches $\{\mathbf{x}^{(j)}\}_{j\in\mathcal{P}^{(i)}}$ based on the similarity scores $s(i,j)$, where $\mathcal{P}^{(i)}$ is a set of patch indices of *top-k* patches in $\mathcal{N}^{(i)}$. We use $k = |\mathcal{P}^{(i)}| = 4$ in our experiments (see Section 4.4 for analysis on the effect of k).

Aggregation module. PASS enforces a query patch $\mathbf{x}^{(i)}$ and its positives $\{\mathbf{x}^{(j)}\}_{j\in\mathcal{P}^{(i)}}$ to be similar as the patch-level self-supervision. To this end, we extract an aggregated representation from the positives by utilizing the [CLS] token, which already has a role for aggregating information in all the patches $\{\mathbf{x}_i\}_{i=1}^N$ (Dosovitskiy et al., 2020). We add an aggregation module h_θ, which is also called class-attention module (Touvron et al., 2021), after f_θ to output an aggregated representation $h_\theta(f_\theta(\mathbf{x}))$. By following the implementation of Touvron et al. (2021), we separate the [CLS] token from the front of f_θ to h_θ. In other words, the input sequence of f_θ becomes $[\mathbf{e}^{(1)}; \mathbf{e}^{(2)}; \dots; \mathbf{e}^{(N)}]$, and the input representation sequence of h_θ is $[\mathbf{e}^{[\texttt{CLS}]}; f_\theta^{(1)}(\mathbf{x}); f_\theta^{(2)}(\mathbf{x}); \dots; f_\theta^{(N)}(\mathbf{x})]$. Then, the output of the aggregation module $h_\theta(f_\theta(\mathbf{x}))$ is:

$$h_\theta(f_\theta(\mathbf{x})) := h_\theta(\mathbf{e}^{[\texttt{CLS}]}, \{f_\theta^{(j)}(\mathbf{x})\}_{j=1,\dots,N}) = h_\theta^{[\texttt{CLS}]}(\mathbf{x}). \tag{3}$$

Also, we denote $h_\theta^{\mathcal{P}^{(i)}}(\mathbf{x}) := h_\theta(\mathbf{e}^{[\texttt{CLS}]}, \{f_\theta^{(j)}(\mathbf{x})\}_{j\in\mathcal{P}^{(i)}})$ as the aggregated representation of positives $\{f_\theta^{(j)}(\mathbf{x})\}_{j\in\mathcal{P}^{(i)}}$ for the given query $\mathbf{x}^{(i)}$.

Training objective. The generic form of the PASS objective can be written as follows:

$$\mathcal{L}^{\text{PASS}}(\mathbf{x};\theta,\tilde{\theta}) := \frac{1}{N}\sum_{i=1}^{N} D(g_\theta(f_\theta^{(i)}(\mathbf{x})), \texttt{sg}(g_{\tilde{\theta}}(h_{\tilde{\theta}}^{\mathcal{P}^{(i)}}(\mathbf{x})))), \tag{4}$$

where D is a distance function and sg is the stop-gradient operation. We found that the aggregation module h is crucial for generating better self-supervision (see Section 4.4 for ablation experiments). Then, the overall training objective is defined as below:

$$\mathcal{L}^{\text{total}}(\{\mathbf{x},\mathbf{x}^+\};\theta,\tilde{\theta}) = \mathcal{L}^{\text{SSL}}(\{\mathbf{x},\mathbf{x}^+\};\theta,\tilde{\theta}) + \lambda\mathcal{L}^{\text{PASS}}(\mathbf{x};\theta,\tilde{\theta}) \tag{5}$$

where λ is a hyperparameter. Note that we use separate projections heads (*i.e.*, non-shared weights) for $\mathcal{L}^{\text{SSL}}$ and $\mathcal{L}^{\text{PASS}}$, respectively. The ov erall training scheme is also illustrated in Figure 2.

3 Related Works

Transformer-like architectures for vision tasks. Vision Transformer (ViT; Dosovitskiy et al. 2020) is the pioneering architecture built on top of Transformer (Vaswani et al., 2017) for vision tasks such as image classification. While Dosovitskiy et al. (2020) pre-train ViTs on a large-scale dataset like JFT300M (Sun et al., 2017) to achieve a high accuracy, Touvron et al. (2020) introduce several optimization strategies tailored to ViTs to be more data-efficient; for example, they achieve competitive performance on ImageNet (Deng et al., 2009) without utilizing external data compared to existing convolutional neural networks (CNNs) such as EfficientNet (Tan & Le, 2019). Inspired by the success of the Vision Transformers, a number of variants (Graham et al., 2021; Heo et al., 2021; Liu et al., 2021; Pan et al., 2021; Wang et al., 2021a; Wu et al., 2021; Zhang et al., 2021) have been developed. They commonly incorporate convolutional designs into ViTs, *e.g.*, a spatial down-sampling operation (Pan et al., 2021; Wu et al., 2021) or a hierarchical structure that considers various patch sizes (Wang et al., 2021a; Liu et al., 2021).

Self-supervised learning. For learning visual representations from a large number of unlabeled images, self-supervised learning (He et al., 2020; Chen et al., 2020a;b; Grill et al., 2020; Caron et al., 2020; 2021; Chen et al., 2021b; Xie et al., 2021c) has become a remarkable research direction, as it can be effectively transferred to various downstream applications like image classification. Much of

Table 1: Object detection and instance segmentation results on the COCO benchmark (Lin et al., 2014). AP^{bb} and AP^{mk} denote bounding box and mask average precision (AP), respectively. We use publicly available pre-trained models for baselines. † denotes results performed in our codebase.

				Detection			Segmentation		
Method	Backbone	Epoch	Param.(M)	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
MoCo-v2	ResNet50	200	26	38.9	59.2	42.4	35.5	56.2	37.8
SwAV	ResNet50	200	26	38.5	60.4	41.4	35.4	57.0	37.7
DenseCL	ResNet50	200	26	40.3	59.9	44.3	36.4	57.0	39.2
ReSim	ResNet50	200	26	40.3	60.6	44.2	36.4	57.5	38.9
DetCo	ResNet50	200	26	40.1	61.0	43.9	36.4	58.0	38.9
DINO†	ViT-S/16	300	22	40.8	63.4	44.2	37.3	59.9	39.5
+ PASS (ours)	ViT-S/16	200	22	**41.5**	**64.1**	**44.8**	**38.0**	**60.8**	**40.3**

the progress comes from exploiting the instance discrimination task (Wu et al., 2018), which learns representations by maximizing the similarity between augmented images originated from the same image and minimizing the similarity between different images. Recently, several approaches (Grill et al., 2020; Caron et al., 2020; 2021) have also shown successful results without using negative pairs; for example, Grill et al. (2020) enforces two representations of a positive pair to be similar with an asymmetric architecture. On the other hand, Pinheiro et al. (2020); Wang et al. (2021b); Xie et al. (2021a); Xiao et al. (2021) propose self-supervised leaning frameworks for dense prediction tasks. Their common design is to match regions between two augmented images, *e.g.,* regions have the same location, while our method matches neighboring regions (i.e., image patches) within the same image for all areas.

4 Experiments

In this section, we demonstrate the effectiveness of the proposed self-supervised learning framework, patch-aware self-supervision (PASS), through extensive large-scale experiments. Specifically, we compare PASS with existing SSL frameworks in various dense prediction benchmarks: (a) COCO object detection and segmentation (Section 4.1), (b) ADE20K segmentation (Section 4.2), and (c) DAVIS video segmentation (Section 4.3). The details of experimental setups are described in each section and Appendix B. We also provide ablation experiments to verify the contribution of each component of our framework in Section 4.4.

Baselines. We consider a variety of existing SSL frameworks developed for ViT (Touvron et al., 2020) and ResNet (He et al., 2016) architectures: (a) self-supervised ViTs: DINO (Caron et al., 2021) and MoCo-v3 (Chen et al., 2021b); and (b) self-supervised ResNets: MoCo-v2 (Chen et al., 2020b), SwAV (Caron et al., 2020), DenseCL (Wang et al., 2021b), ReSim (Xiao et al., 2021) and DetCo (Xie et al., 2021a). We use ViT-S/16 (22M parameters) and ResNet50 (26M parameters) since they are conventional choices and have the similar number of parameters. We denote our method built upon an existing method by "+ PASS", *e.g.,* DINO + PASS.

Implementation details. For pre-training, we incorporate the PASS objective with the state-of-the-art SSL framework, DINO (Caron et al., 2021), as described in (5). We pre-train ViT-S/16 (Touvron et al., 2020) on ImageNet (Deng et al., 2009) for 200 epochs with a batch size of 1024. We follow DINO's training details (*e.g.*, optimizer, learning rate schedule). For the projection head of the PASS objective, we follow the architecture of DINO except the final dimension $K = 4096$. For the distance function D, we use the KL divergence in $\mathcal{L}^{\text{PASS}}$ (4) following DINO. For our aggregation module, we follow the implementation of class-attention (Touvron et al., 2021) without their normalization technique. The other training details are provided in Appendix A.

4.1 COCO object detection and segmentation

Setup. We evaluate pre-trained models on the COCO object detection and instance segmentation tasks (Lin et al., 2014). Specifically, all models are fine-tuned on the COCO `train2017` split with

Table 2: (a) Semantic segmentation on ADE20K (Zhou et al., 2017), and (b) video object segmentation on DAVIS 2017 (Pont-Tuset et al., 2017). We use publicly available pre-trained models for baselines. All results are performed in our codebase.

	(a) ADE20K						(b) DAVIS 2017		
Method	Backbone	Epoch	Param.(M)	mIoU	aAcc	mAcc	$(\mathcal{J}\&\mathcal{F})_m$	$\mathcal{J}_m$	$\mathcal{F}_m$
MoCo-v2	ResNet50	200	26	35.76	77.63	45.08	55.5	56.0	55.0
SwAV	ResNet50	200	26	35.40	77.49	44.92	57.4	57.6	57.3
DenseCL	ResNet50	200	26	37.19	78.53	47.08	50.7	52.6	48.9
ReSim	ResNet50	200	26	36.61	78.38	46.36	49.3	51.2	47.3
DetCo	ResNet50	200	26	37.28	78.42	46.73	56.7	57.0	56.4
DINO	ViT-S/16	300	22	31.13	75.98	41.43	60.7	59.1	62.4
+ PASS (ours)	ViT-S/16	200	22	**38.18**	**79.42**	**49.31**	**61.4**	**59.7**	**63.1**

the standard `1x` schedule, and then evaluated on the COCO `val2017` split. To perform detection and segmentation, we use Mask R-CNN (He et al., 2017) with FPN (Lin et al., 2017). We follow the fine-tuning details of El-Nouby et al. (2021) including the optimizer and the FPN architecture.

Results. Table 1 shows that our PASS consistently improves DINO in both detection and segmentation tasks, and consequently, DINO + PASS outperforms all the baselines. For example, the bounding box average precision (*i.e.*, AP^{bb}) of DINO + PASS is 0.7 points higher than that of DINO. One can find similar results in the segmentation task; for example, DINO + PASS achieves 38.0 mask average precision (*i.e.*, AP^{mk}), which is 0.7 points higher than DINO, and also 1.6 points higher than the best ResNet-based baseline, DenseCL (Wang et al., 2021b). We emphasize that the improvements from our framework are even achieved with a smaller number of pre-training epochs (*i.e.,* 200 epochs). These results demonstrate that the advantages of our framework are not only high performance, but also training efficiency.

4.2 ADE20K SEMANTIC SEGMENTATION

Setup. We evaluate segmentation performance of pre-trained models on the ADE20K (Zhou et al., 2017) benchmark, which contains 150 fine-grained semantic categories and 25k training images. All models are fine-tuned with Semantic FPN (Kirillov et al., 2019) under the standard 40k iteration schedule. We follow the training details of Contributors (2020). We report three evaluation metrics: (a) mean Intersection of Union (mIoU) averaged over all semantic categories, (b) all pixel accuracy (aAcc), and (c) mean accuracy of each class (mAcc).

Results. As shown in Table 2(a), DINO + PASS achieves significant improvements over DINO in all the metrics; for example, DINO + PASS achieves 7.05 and 7.88 points higher than DINO in terms of the mIoU and mAcc metrics, respectively. Also, DINO + PASS consistently outperforms all the CNN-based SSL baselines; for example, in terms of the mIoU metric, our method achieves 38.18 point, while DetCO and DenseCL do 37.28 and 37.19 points, respectively. These comparisons across the architectures verify the effectiveness of PASS.

4.3 DAVIS VIDEO OBJECT SEGMENTATION

Setup. We evaluate video object segmentation performance of pre-trained models on the DAVIS 2017 benchmark (Pont-Tuset et al., 2017). We follow the experimental protocol in Jabri et al. (2020); Caron et al. (2021), which does not require training costs. To be specific, it evaluates the quality of frozen representations of image patches by segmenting scenes with a nearest neighbor between consecutive frames. We report two evaluation metrics of mean region similarity $\mathcal{J}_m$, and mean contour-based accuracy $\mathcal{F}_m$. We also report their average score $(\mathcal{J}\&\mathcal{F})_m$.

Results. In Table 2(b), DINO + PASS not only consistently improves DINO, but also largely surpass the other baselines. For example, PASS improves the $\mathcal{F}_m$ score of DINO from 62.4 to 63.1, while DenseCL and ReSim achieve only 48.9 and 47.3, respectively. We present visualizations of video object segmentation results obtained by DINO and DINO + PASS in Figure 4. As shown in the figure, PASS clearly enhances the video segmentation quality. Overall, we observe that our method

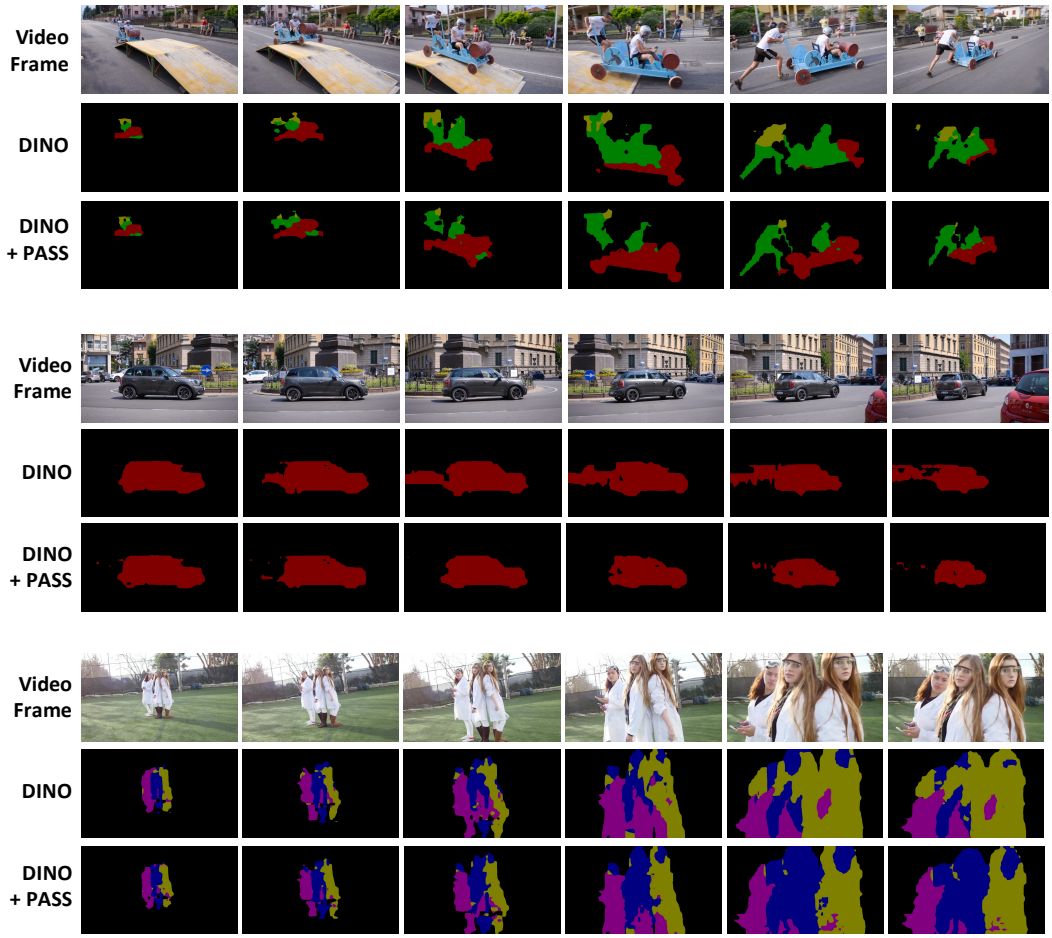

Figure 4: Visualization of segmentation results. **Top**: input video frames. **Middle** and **Bottom**: segmentation results obtained by DINO and DINO + PASS (ours), respectively. PASS clearly improves the segmentation results, which is an evidence that PASS encourages the patch representations to learn semantic information of each object.

encourages the model to produce more meaningful segmentation maps, which also demonstrate the effectiveness of our framework in learning better patch-level representations.

4.4 Ablation Study

We perform ablation studies to further understand how PASS works. Specifically, we assess the individual effects of PASS's components and show that each of them has an orthogonal contribution to the overall improvements. To this end, we pre-train ViT-Ti/16 on the MS COCO `train2007` dataset for 200 epochs with a batch size of 256. Table 3 summarizes the results of semantic segmentation performance on the ADE20K benchmark (Zhou et al., 2017).

Effect of neighboring patches. We demonstrate the effect of considering neighboring patches as positive candidates. Without the consideration (*i.e.*, "w/o Neighbors"), we use the entire patches as the candidates. As shown in the the third and fourth rows in Table 3(a), selecting positives from the neighboring patches shows better performance (*e.g.*, 25.83 mIoU) on the segmentation task compared to considering the entire patches (*e.g.*, 22.03 mIoU). This result verifies that positive patches tend to be adjacent as we expected.

Effect of positive matching. Our positive matching module selects *top-k* patches as positives from the neighborhood using the cosine similarity on the representation space. To demonstrate the effect of this module, we use all neighboring patches as positives (*i.e.*, w/o "Matching" or $k = 8$). As

Table 3: Ablation studies on (a) each contribution of three components in our method: the neighboring patches ("Neighbors"), positive matching ("Matching") and aggregation module ("Aggregation"); and (b) varying the number of positive patches used in positive matching. All models are pre-trained on the COCO benchmark (Lin et al., 2014). We evaluate the pre-trained models using the semantic segmentation benchmark, ADE20K (Zhou et al., 2017).

(a) Ablation study on component contributions

Method	mIoU	aAcc	mAcc
DINO	21.81	72.21	29.61
+ PASS (ours)	**25.83**	**74.41**	**35.04**
w/o "Neighbor"	22.03	72.09	29.96
w/o "Matching"	9.49	61.63	13.14
w/o "Aggregation"	16.04	68.16	22.28

(b) Ablation study on "Matching"

# positives	mIoU	aAcc	mAcc
8	9.49	61.63	13.14
4	**25.83**	**74.41**	**35.04**
2	23.59	73.10	31.76
1	9.51	61.82	13.19

shown in the fifth row in Table 3, we found that aggregating all neighboring patches is harmful for learning representations. Table 3(b) also shows the effect of k. We empirically found that using only the *top-1* patch (*i.e.*, $k = 1$) is also not effective, and aggregating only few patches ($k = 2$ or 4) is essential for the performance. As observed in Table 3(b), we select $k = 4$ patches among the neighborhood in the positive matching module.

Effect of aggregation module. We here validate the contribution of our aggregation module ("Aggregation") which aims at aggregating multiple patch representations. To this end, we simply replace the aggregation module by the average pooling operation. The last row in Table 3(a) shows the performance of DINO + PASS without "Aggregation", which underperforms the baseline, DINO. This is an evidence that our attention-based aggregation module is a crucial component of our framework.

5 Conclusion

We propose *Patch-Aware Self-Supervision* (PASS), a simple yet effective self-supervised learning framework for learning visual representations of individual image patches. Our key idea is to treat semantically similar neighboring patches as positives. Specifically, we select semantically similar (*i.e.*, positive) patches from the neighborhood, and then enforce invariance against each patch and its positives. Through the extensive experiments, we demonstrate the effectiveness of our framework in various downstream tasks, including object detection and semantic segmentation. We believe that this work would guide many research directions for self-supervised learning.

Ethics Statement

Due to the absence of supervision, self-supervised learning often requires a vast number of training samples to obtain meaningful representations, for example, Codex (Chen et al., 2021a) is trained on 159GB python code collected from public repositories in GitHub. Here, the data collection process may cause unexpected social issues, *e.g.*, privacy infringement, because it is impossible to check all the data in person. In this respect, researchers are responsible to develop *data-efficient* self-supervised learning schemes. We believe that designing a new type of self-supervision (*e.g.*, patch-level instead of image-level self-supervision) would be a promising research direction towards the data-efficient self-supervised learning.

Reproducibility Statement

We describe the additional training details of the model in Appendix A, and evaluation details in Appendix B. One can find our reproducible code in the supplementary material.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pp. 1877–1901, 2020.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer, 2020.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021a. URL https://arxiv.org/abs/2107.03374.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020a.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021b.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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. *arXiv preprint arXiv:2010.11929*, 2020.

Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681*, 2021.

Ben Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. *arXiv preprint arXiv:2104.01136*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. *arXiv preprint arXiv:2103.16302*, 2021.

Allan Jabri, Andrew Owens, and Alexei A Efros. Space-time correspondence as a contrastive random walk. *arXiv preprint arXiv:2006.14613*, 2020.

Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6399–6408, 2019.

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*, 2014.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

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. *arXiv preprint arXiv:2103.14030*, 2021.

Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable visual transformers with hierarchical pooling. *arXiv preprint arXiv:2103.10619*, 2021.

Pedro O Pinheiro, Amjad Almahairi, Ryan Y Benmalek, Florian Golemo, and Aaron Courville. Unsupervised learning of dense visual representations. In *NeurIPS*, 2020.

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

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.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pp. 843–852, 2017.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

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*, 2021a.

Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3024–3033, 2021b.

Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. In *CVPR*, 2018.

Tete Xiao, Colorado J Reed, Xiaolong Wang, Kurt Keutzer, and Trevor Darrell. Region similarity representation learning. In *ICCV*, 2021.

Enze Xie, Jian Ding, Wenhai Wang, Xiaohang Zhan, Hang Xu, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. In *ICCV*, 2021a.

Enze Xie, Wenjia Wang, Wenhai Wang, Peize Sun, Hang Xu, Ding Liang, and Ping Luo. Segmenting transparent object in the wild with transformer. *arXiv preprint arXiv:2101.08461*, 2021b.

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*, 2021c.

Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *arXiv preprint arXiv:2103.15358*, 2021.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.

A TRAINING SETUP

A.1 TRAINING DATASETS

We use the ImageNet (Deng et al., 2009) dataset for large-scale pre-training, and the MS COCO (Lin et al., 2014) dataset with `train2007` split for medium-scale pre-training.

ImageNet (Deng et al., 2009) is a 1000-class natural image dataset that contains 1,281,167 training images, 50,000 validation images. The dataset can be downloaded at https://www.image-net.org/challenges/LSVRC/index.php.

COCO (Lin et al., 2014) is a large-scale object detection, segmentation, and captioning dataset published by Microsoft. We use `train2007` split for pre-training in Sec 4.4. The dataset can be downloaded at https://cocodataset.org/#download.

A.2 TRAINING DETAILS

We pre-train ViT-S/16 (Touvron et al., 2020) on the ImageNet (Deng et al., 2009) dataset for 200 training epochs with a batch size of 1024. In the case of the joint usage of DINO (Caron et al., 2021) and our framework PASS ("DINO + PASS"), we follow details of DINO, which also is available at https://github.com/facebookresearch/dino. We specify several details as follows:

- **Projection head.** Caron et al. (2021) uses a single hyperparameter K for the final output dimension. For the SSL projection head, we use $K = 65536$, and $K = 4096$ for the PASS projection head.
- **Momentum encoder.** We use a momentum value of exponential moving average to 0.996.
- **Teacher temperature.** Caron et al. (2021) uses three hyperparameters for the intial and final values and warmup epochs for teacher temperature. We use 0.04 as the initial, 0.07 to the final and 30 epochs for warmup.
- **Gradient clipping.** We use the maximal value of gradient norm to be 3.0.
- **Optimizer.** We use adamw optimizer with learning rate of 0.0005 under linear scaling rule. We also use cosine scheduling with minimum learning rate of 0.00001. For weight decay, we use 0.04 as the initial, 0.4 to the final.
- **Multi-crop augmentation.** We use (0.25, 1.) of global crop scale and (0.05, 0.25) of local crop scale. We use 2 global crops and 2 local crops.
- **Aggregation module.** We use two lengths of class-attention blocks (Touvron et al., 2021) without Layerscale normalization (Touvron et al., 2021) as our aggregation module.
- **Loss weight.** We use our loss weight $\lambda = 0.1$.

We use the same hyperparameters when we pre-train ViT-Ti/16 on the COCO dataset in Table 3, except a batch size of 256, $K = 4096$ for the SSL projection head.

B EVALUATION SETUP

B.1 EVALUATION DATASETS

For evaluation, we use the MS COCO (Lin et al., 2014), the ADE20K (Zhou et al., 2017), and the DAVIS-2017 benchmarks.

ADE20K (Zhou et al., 2017) is a semantic segmentation dataset contains 150 fine-grained semantic categories and 25k images. The dataset can be downloaded at http://groups.csail.mit.edu/vision/datasets/ADE20K/toolkit/index_ade20k.pkl.

DAVIS 2017 (Pont-Tuset et al., 2017) is a dataset for video object segmentation. It contains a total of 150 videos that consists of 60 training, 30 validation, and 60 testing videos. We use 480p image resolution. The dataset can be downloaded at https://davischallenge.org/davis2017/code.html.

B.2 EVALUATION DETAIL

We evaluate transferring performances of pre-trained models to various downstream tasks as follows:

- **COCO object detection and instance segmentation.** We follow the basic configuration of `mmdetection`((Chen et al., 2019), which is available at https://github.com/open-mmlab/mmdetection for fine-tuning Mask R-CNN (He et al., 2017) with FPN (Lin et al., 2017) and `1x` schedule. In addition, we adapt augmentation, optimizer and FPN architecture for ViT-S/16 architecture from (El-Nouby et al., 2021), which is available at https://github.com/facebookresearch/xcit/tree/main/detection.
- **ADE20K semantic segmentation.** We follow the all configuration of `mmsegmentation` (Contributors (2020); https://github.com/open-mmlab/mmsegmentation) for fine-tuning Semantic FPN (Kirillov et al., 2019) with `40k` iterations, as it supports ViT-S/16 architecture.
- **DAVIS 2017 video object segmentation.** We follow evaluation protocol of Caron et al. (2021), which is available at https://github.com/facebookresearch/dino.