Propagate Yourself: Exploring Pixel-Level Consistency for Unsupervised Visual Representation Learning

Zhenda Xie*[13], Yutong Lin*[23], Zheng Zhang[3], Yue Cao[3], Stephen Lin[3], Han Hu[3]
[1]Tsinghua University [2]Xi'an Jiaotong University
[3]Microsoft Research Asia
xzd18@mails.tsinghua.edu.cn yutonglin@stu.xjtu.edu.cn
{zhez,yuecao,stevelin,hanhu}@microsoft.com

Abstract

Contrastive learning methods for unsupervised visual representation learning have reached remarkable levels of transfer performance. We argue that the power of contrastive learning has yet to be fully unleashed, as current methods are trained only on instance-level pretext tasks, leading to representations that may be sub-optimal for downstream tasks requiring dense pixel predictions. In this paper, we introduce pixel-level pretext tasks for learning dense feature representations. The first task directly applies contrastive learning at the pixel level. We additionally propose a pixel-to-propagation consistency task that produces better results, even surpassing the state-of-the-art approaches by a large margin. Specifically, it achieves ***60.2*** *AP,* ***41.4 / 40.5*** *mAP and* ***77.2*** *mIoU when transferred to Pascal VOC object detection (C4), COCO object detection (FPN / C4) and Cityscapes semantic segmentation using a ResNet-50 backbone network, which are* ***2.6*** *AP,* ***0.8 / 1.0*** *mAP and* ***1.0*** *mIoU better than the previous best methods built on instance-level contrastive learning. Moreover, the pixel-level pretext tasks are found to be effective for pre-training not only regular backbone networks but also head networks used for dense downstream tasks, and are complementary to instance-level contrastive methods. These results demonstrate the strong potential of defining pretext tasks at the pixel level, and suggest a new path forward in unsupervised visual representation learning. Code is available at* https://github.com/zdaxie/PixPro.

1. Introduction

According to Yann LeCun, "if intelligence is a cake, the bulk of the cake is unsupervised learning". This quote reflects his belief that human understanding of the world is predominantly learned from the tremendous amount of unlabeled information within it. Research in machine intelligence has increasingly moved in this direction, with substantial progress in unsupervised and self-supervised learning[34, 18, 25, 8, 30]. In computer vision, recent advances can largely be ascribed to the use of a pretext task called *instance discrimination*, which treats each image in a training set as a single class and aims to learn a feature representation that discriminates among all the classes.

Although self-supervised learning has proven to be remarkably successful, we argue that there remains significant

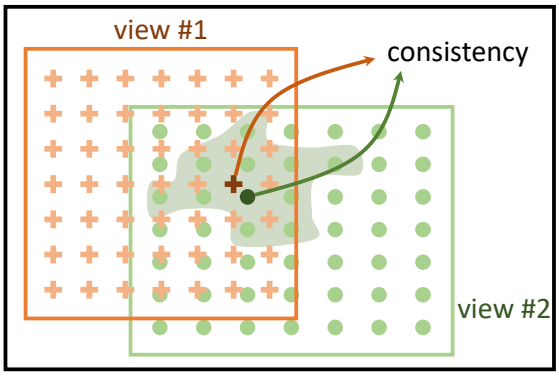

Figure 1. An illustration of the proposed *PixPro* method, which is based on a *pixel-to-propagation consistency* pretext task for pixel-level visual representation learning. In this method, two views are randomly cropped from an image (outlined in black), and the features from the corresponding pixels of the two views are encouraged to be consistent. For one of them, the feature comes from a regular pixel representation (illustrated as orange crosses). The other feature comes from a smoothed pixel representation (shown as green dots) built by propagating the features of similar pixels (illustrated as the light green region). Note that this hard selection of similar pixels is for illustration only. In implementation, all pixels on the same view will contribute to propagation, with the propagation weight of each pixel determined by its feature similarity to the center pixel.

*Equal Contribution. The work is done when Zhenda Xie and Yutong Lin are interns at Microsoft Research Asia.

untapped potential. The self-supervision that guides representation learning in current methods is based on image-level comparisons. As a result, the pre-trained representation may be well-suited for image-level inference, such as image classification, but may lack the spatial sensitivity needed for downstream tasks that require pixel-level predictions, e.g., object detection and semantic segmentation. How to perform self-supervised representation learning at the pixel level is a problem that until now has been relatively unexplored.

In this paper, we tackle this problem by introducing pixel-level pretext tasks for self-supervised visual representation learning. Inspired by recent instance discrimination methods, our first attempt is to construct a pixel-level contrastive learning task, where each pixel in an image is treated as a single class and the goal is to distinguish each pixel from others within the image. Features from the same pixel are extracted via two random image crops containing the pixel, and these features are used to form positive training pairs. On the other hand, features obtained from different pixels are treated as negative pairs. With training data collected in this self-supervised manner, a contrastive loss is applied to learn the representation. We refer to this approach as *PixContrast*.

In addition to this contrastive approach, we present a method based on *pixel-to-propagation consistency*, where positive pairs are obtained by extracting features from the same pixel through two asymmetric pipelines instead. The first pipeline is a standard backbone network with a projection head. The other has a similar form but ends with a proposed pixel propagation module, which filters the pixel's features by propagating the features of similar pixels to it. This filtering introduces a certain smoothing effect, while the standard feature maintains spatial sensitivity. A difference of this method from the contrastive approach of *PixContrast* is that it encourages consistency between positive pairs without consideration of negative pairs. While the performance of contrastive learning is known to be influenced heavily by how negative pairs are handled [18, 8], this issue is avoided in this consistency-based pretext task. Empirically, we find that this pixel-to-propagation consistency method, which we call *PixPro*, significantly outperforms the *PixContrast* approach over various downstream tasks.

Besides learning good pixel-level representations, the proposed pixel-level pretext tasks are found to be effective for pre-training on not only backbone networks but also head networks used for dense downstream tasks, contrary to instance-level discrimination where only backbone networks are pre-trained and transferred. This is especially beneficial for downstream tasks with limited annotated data, as all layers can be well-initialized. Moreover, the proposed pixel-level approach is complementary to existing instance-level methods, where the former is good at learning a spatially sensitive representation and the latter provides better categorization ability. A combination of the two methods capitalizes on both of their strengths, while also remaining computationally efficient in pre-training as they both can share a data loader and backbone encoders.

The proposed *PixPro* achieves state-of-the-art transfer performance on common downstream benchmarks requiring dense prediction. Specifically, with a ResNet-50 backbone, it obtains 60.2 AP on Pascal VOC object detection using a Faster R-CNN detector (C4 version), 41.4 / 40.5 mAP on COCO object detection using a Mask R-CNN detector (both the FPN / C4 versions, $1\times$ settings), and 77.2 mIoU Cityscapes semantic segmentation using an FCN method, which are 2.6 AP, 0.8 / 1.0 mAP, and 1.0 mIoU better than the leading unsupervised/supervised methods. Though past evaluations of unsupervised representation learning have mostly been biased towards linear classification on ImageNet, we advocate a shift in attention to performance on downstream tasks, which is the main purpose of unsupervised representation learning and a promising setting for pixel-level approaches.

2. Related Works

Instance discrimination Unsupervised visual representation learning is currently dominated by the pretext task of instance discrimination, which treats each image as a single class and learns representations by distinguishing each image from all the others. This line of investigation can be traced back to [14], and after years of progress [34, 29, 21, 38, 1, 35], transfer performance superior to supervised methods was achieved by MoCo [18] on a broad range of downstream tasks. After this milestone, considerable attention has been focused in this direction [8, 30, 3, 17, 5]. While follow-up works have quickly improved the linear evaluation accuracy (top-1) on ImageNet-1K from about 60% [18] to higher than 75% [5] using a ResNet-50 backbone, the improvements on downstream tasks such as object detection on Pascal VOC and COCO have been marginal.

Instead of using instance-level pretext tasks, our work explores pretext tasks at the pixel level for unsupervised feature learning. We focus on transfer performance to downstream tasks such as object detection and semantic segmentation, which have received limited consideration in prior research. We show that pixel-level representation learning can surpass the existing instance-level methods by a significant margin, demonstrating the potential of this direction.

Other pretext tasks using a single image Aside from instance discrimination, there exist numerous other pretext tasks including context prediction [13], grayscale image colorization [36], jigsaw puzzle solving [26], split-brain auto-encoding [37], rotation prediction [16], learning to

cluster [4], and missing part prediction [21, 32, 7]. Interest in these tasks for unsupervised feature learning has fallen off considerably due to their inferior performance and greater complexity in architectures or training strategies. Among these methods, the approach most related to ours is missing parts prediction [21, 32, 7], which was inspired by successful pretext tasks in natural language processing [12, 2]. Like our pixel-propagation consistency technique, such methods also operate locally. However, they either partition images into patches [32, 21] or require special architectures/training strategies to perform well [21, 7], while our approach directly operates on pixels and has no special requirements on the encoding networks. Training with our method is also simple, with few bells and whistles. More importantly, our approach achieves state-of-the-art transfer performance on the important dense prediction tasks of object detection and semantic segmentation.

3. Method

3.1. Pixel-level Contrastive Learning

The state-of-the-art unsupervised representation learning methods are all built on the pretext task of instance discrimination. In this section, we show that the idea of instance discrimination can be also applied at the pixel level for learning visual representations that generalize well to downstream tasks. We adopt the prevalent contrastive loss to instantiate the pixel-level discrimination task, and call this method *PixContrast*.

As done in most instance-level contrastive learning methods, *PixContrast* starts by sampling two augmentation views from the same image. The two views are both resized to a fixed resolution (e.g., 224×224) and are passed through a regular encoder network and a momentum encoder network [18, 9, 17] to compute image features. The encoder networks are composed of a backbone network and a projection head network, where the former could be any image neural network (we adopt ResNet by default), and the latter consists of two successive 1×1 convolution layers (of 2048 and 256 output channels, respectively) with a batch normalization layer and a ReLU layer in-between to produce image feature maps of a certain spatial resolution, e.g., 7×7. While previous methods compute a single image feature vector for each augmentation view, *PixContrast* computes a feature map upon which pixel-level pretext tasks can be applied. The learnt backbone representations are then used for feature transfer. An illustration of the architecture is shown in Figure 2.

Pixel Contrast With the two feature maps computed from two views, we can construct pixel contrast pretext tasks for representation learning. Each pixel in a feature map is first warped to the original image space, and the distances between all pairs of pixels from the two feature maps are computed. The distances are normalized to the diagonal length of a feature map bin to account for differences in scale between the augmentation views, and the normalized distances are used to generate positive and negative pairs, based on a threshold $\mathcal{T}$:

$$A(i,j) = \begin{cases} 1, & \text{if dist}(i,j) \leq \mathcal{T}, \\ 0, & \text{if dist}(i,j) > \mathcal{T}, \end{cases} \tag{1}$$

where i and j are pixels from each of the two views; $\text{dist}(i,j)$ denotes the normalized distance between pixel i and j in the original image space; and the threshold is set to $\mathcal{T} = 0.7$ by default.

Similar to instance-level contrastive learning methods, we adopt a contrastive loss for representation learning:

$$\mathcal{L}_{\text{Pix}}(i) = -\log \frac{\sum_{j \in \Omega_p^i} e^{\cos(\mathbf{x}_i, \mathbf{x}'_j)/\tau}}{\sum_{j \in \Omega_p^i} e^{\cos(\mathbf{x}_i, \mathbf{x}'_j)/\tau} + \sum_{k \in \Omega_n^i} e^{\cos(\mathbf{x}_i, \mathbf{x}'_k)/\tau}}, \tag{2}$$

where i is a pixel in the first view that is also located in the second view; Ω_p^i and Ω_n^i are sets of pixels in the second view assigned as positive and negative, respectively, with respect to pixel i; $\mathbf{x}_i$ and $\mathbf{x}'_j$ are the pixel feature vectors in two views; and τ is a scalar temperature hyper-parameter, set by default to 0.3. The loss is averaged over all pixels on the first view that lie in the intersection of the two views. Similarly, the contrastive loss for a pixel j on the second view is also computed and averaged. The final loss is the average over all image pairs in a mini-batch.

As later shown in the experiments, this direct extension of instance-level contrastive learning to the pixel level performs well in representation learning.

3.2. Pixel-to-Propagation Consistency

The *spatial sensitivity* and *spatial smoothness* of a learnt representation may affect transfer performance on downstream tasks requiring dense prediction. The former measures the ability to discriminate spatially close pixels, needed for accurate prediction in boundary areas where labels change. The latter property encourages spatially close pixels to be similar, which can aid prediction in areas that belong to the same label. The *PixContrast* method described in the last subsection only encourages the learnt representation to be *spatially sensitive*. In the following, we present a new pixel-level pretext task which additionally introduces *spatial smoothness* in the representation learning.

This new pretext task involves two critical components. The first is a pixel propagation module, which filters a pixel's features by propagating the features of similar pixels to it. This propagation has a feature denoising/smoothing effect on the learned representation that leads to more coherent solutions among pixels in pixel-level prediction tasks.

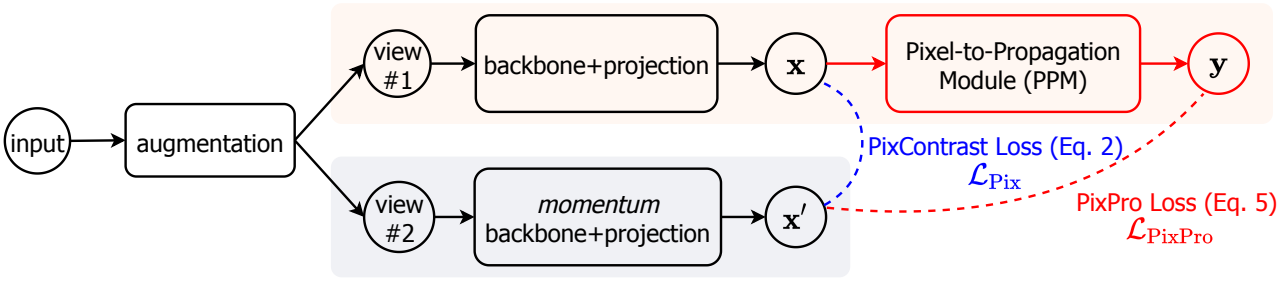

Figure 2. Architecture of the *PixContrast* and *PixPro* methods.

The second component is an asymmetric architecture design where one branch produces a regular feature map and the other branch incorporates the pixel-propagation module. The pretext task seeks consistency between the features from the two branches without considering negative pairs. On the one hand, this design maintains the *spatial sensitivity* of the learnt representation to some extent, thanks to the regular branch. On the other hand, while the performance of contrastive learning is known to be heavily affected by the treatment of negative pairs [18, 8], the asymmetric design enables the representation learning to rely only on consistency between positive pairs, without facing the issue of carefully tuning negative pairs [17]. We refer to this pretext task as *pixel-to-propagation consistency (PPC)* and describe these primary components in the following.

Pixel Propagation Module For each pixel feature $\mathbf{x}_i$, the pixel propagation module computes its smoothed transform $\mathbf{y}_i$ by propagating features from all pixels $\mathbf{x}_j$ within the same image Ω as

$$\mathbf{y}_i = \Sigma_{j\in\Omega} s(\mathbf{x}_i, \mathbf{x}_j) \cdot g(\mathbf{x}_j), \tag{3}$$

where $s(\cdot,\cdot)$ is a similarity function defined as

$$s(\mathbf{x}_i, \mathbf{x}_j) = (\max(\cos(\mathbf{x}_i, \mathbf{x}_j), 0))^{\gamma}, \tag{4}$$

with γ being an exponent to control the sharpness of the similarity function and is set by default to 2; $g(\cdot)$ is a transformation function that can be instantiated by l linear layers with a batch normalization and a ReLU layer between two successive layers. When $l = 0$, $g(\cdot)$ is an identity function and Eq. (3) will be a non-parametric module. Empirically, we find that all of $l = \{0, 1, 2\}$ perform well, and we set $l = 1$ by default as its results are slightly better. Figure 3 illustrates the proposed pixel propagation module.

Pixel-to-Propagation Consistency Loss In the asymmetric architecture design, there are two different encoders: a regular encoder with the pixel propagation module applied afterwards to produce smoothed features, and a momentum encoder without the propagation module. The two augmentation views both pass through the two encoders, and the features from different encoders are encouraged to be consistent:

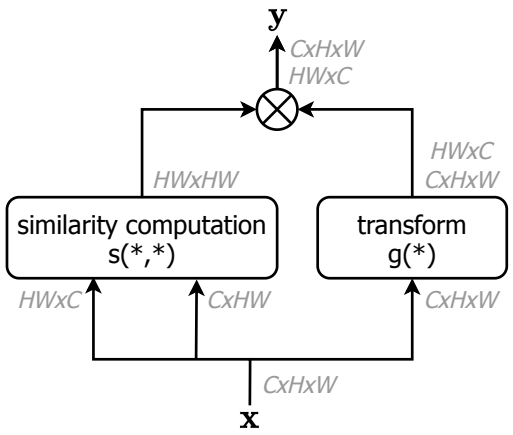

Figure 3. Illustration of the pixel propagation module (*PPM*). The input and output resolutions of each computation block are included.

$$\mathcal{L}_{\text{PixPro}} = -\cos(\mathbf{y}_i, \mathbf{x}'_j) - \cos(\mathbf{y}_j, \mathbf{x}'_i), \tag{5}$$

where i and j are a positive pixel pair from two augmentation views according to the assignment rule in Eq. (1); $\mathbf{x}'_i$ and $\mathbf{y}_i$ are pixel feature vectors of the momentum encoder and the propagation encoder, respectively. This loss is averaged over all positive pairs for each image, and then further averaged over images in a mini-batch to drive the representation learning.

Comparison to *PixContrast* The overall architecture of the pixel-to-propagation consistency (*PPC*) method is illustrated in Figure 2. Compared to the *PixContrast* method described in Section 3.1 (see the blue-color loss in Figure 2), there are two differences: the introduction of a pixel propagation module (*PPM*), and replacement of the contrastive loss by a consistency loss. Table 2(c) and 3 show that both changes are critical for the feature transfer performance.

Computation complexity The proposed *PixContrast* and *PixPro* approaches adopt the same data loader and back-

bone architectures as those of the instance discrimination based representation learning methods. There computation complexity in pre-training is thus similar as that of the counterpart instance-level method (i.e. BYOL [17]): 8.6G vs. 8.2G FLOPs using a ResNet-50 backbone architecture, where the head and loss contribute about 0.4G FLOPs overhead.

3.3. Aligning Pre-training to Downstream Networks

Previous visual feature pre-training methods are generally limited to classification backbones. For supervised pre-training, i.e. by the ImageNet image classification pretext task, the standard practice is to transfer only the pre-trained backbone features to downstream tasks. The recent unsupervised pre-training methods have continued this practice. One reason is that the pre-training methods operate at the instance level, making them less compatible with the dense prediction required in head networks for downstream tasks.

In contrast, the fine-grained spatial inference of pixel-level pretext tasks more naturally aligns with dense downstream tasks. To examine this, we consider an object detection method, FCOS [31], for dense COCO detection. FCOS [31] applies a feature pyramid network (FPN) from P3 ($8\times$ down-sampling) to P7 ($128\times$ down-sampling) [22], followed by two separate convolutional head networks (shared for all pyramidal levels) on the output feature maps of a ResNet backbone to produce classification and regression results.

We adopt the same architecture from the input image until the third 3×3 convolutional layer in the head. In FPN, we involve feature maps from P3 to P6, with P7 omitted because the resolution is too low. A pixel propagation module (*PPM*) with shared weights and the *pixel-to-propagation consistency* (*PPC*) loss described in Section 3.2 are applied on each pyramid level. The final loss is first averaged at each pyramidal level and then averaged over all the pyramids.

Pre-training the FPN layers and the head networks used for downstream tasks can generally improve the transfer accuracy, as shown in Tables 5 and 6.

3.4. Combined with Instance Contrast

The presented pixel-level pretext tasks adopt the same data loader and encoders as in state-of-the-art instance-level discrimination methods [18, 17], with two augmentation views sampled from each image and fed into backbone encoders. Hence, our pixel-level methods can be conveniently combined with instance-level pretext tasks, by sharing the same data loader and backbone encoders, with little pre-training overhead.

Specifically, the instance-level pretext task is applied on the output of the *res5* stage, using projection heads that are independent of the pixel-level task. Here, we use a popular instance-level method, SimCLR [8], with a momentum encoder to be aligned with the pixel-level pretext task. In this combination, the two losses from the pixel-level and instance-level pretext tasks are balanced by a multiplicative factor α (set to 1 by default), as

$$\mathcal{L} = \mathcal{L}_{\text{PixPro}} + \alpha \mathcal{L}_{\text{inst}}. \tag{6}$$

In general, the two tasks are complementary to each other: a pixel-level pretext task learns representations good for spatial inference, while an instance-level pretext task is good for learning categorization representations. Table 4 shows that an additional instance-level contrastive loss can significantly improve ImageNet-1K linear evaluation, indicating that a better categorization representation is learnt. Likely because of better categorization ability, it achieves noticeably improved transfer accuracy on the downstream task of FCOS [31] object detection on COCO (about 1 mAP improvement).

4. Experiments

4.1. Pre-training Settings

Datasets We adopt the widely used ImageNet-1K [11] dataset for feature pre-training, which consists of $\sim$1.28 million training images.

Architectures Following recent unsupervised methods [18, 17], we adopt the ResNet-50 [20] model as our backbone network. The two branches use different encoders, with one using a regular backbone network and a regular projection head, and the other using the momentum network with a moving average of the parameters of the regular backbone network and the projection head. The proposed pixel propagation module (*PPM*) is applied on the regular branch.The FPN architecture with P3-P6 feature maps are also tested in some experiments.

Data Augmentation In pre-training, the data augmentation strategy follows [17], where two random crops from the image are independently sampled and resized to 224×224 with a random horizontal flip, followed by color distortion, Gaussian blur, and a solarization operation. We skip the loss computation for cropped pairs with no overlaps, which compose only a small fraction of all the cropped pairs.

Optimization We vary the training length from 50 to 400 epochs, and use 100-epoch training in our ablation study. The *LARS* optimizer with a cosine learning rate scheduler and a base learning rate of 1.0 is adopted in training, where the learning rate is linearly scaled with the batch size as $\text{lr} = \text{lr}_{\text{base}} \times \#\text{bs}/256$. Weight decay is set to 1e-5. The total batch size is set to 1024, using 8 V100 GPUs. For the

Method	#. Epoch	Pascal VOC (R50-C4)			COCO (R50-FPN)			COCO (R50-C4)			Cityscapes (R50)
		AP	AP_{50}	AP_{75}	mAP	AP_{50}	AP_{75}	mAP	AP_{50}	AP_{75}	mIoU
scratch	-	33.8	60.2	33.1	32.8	51.0	35.3	26.4	44.0	27.8	65.3
supervised	100	53.5	81.3	58.8	39.7	59.5	43.3	38.2	58.2	41.2	74.6
MoCo [18]	200	55.9	81.5	62.6	39.4	59.1	43.0	38.5	58.3	41.6	75.3
SimCLR [8]	1000	56.3	81.9	62.5	39.8	59.5	43.6	38.4	58.3	41.6	75.8
MoCo v2 [9]	800	57.6	82.7	64.4	40.4	60.1	44.3	39.5	59.0	42.6	76.2
InfoMin [30]	200	57.6	82.7	64.6	40.6	60.6	44.6	39.0	58.5	42.0	75.6
InfoMin [30]	800	57.5	82.5	64.0	40.4	60.4	44.3	38.8	58.2	41.7	75.6
PixPro (ours)	100	58.8	83.0	66.5	41.3	61.3	45.4	40.0	59.3	43.4	76.8
PixPro (ours)	400	**60.2**	**83.8**	**67.7**	**41.4**	**61.6**	**45.4**	**40.5**	**59.8**	**44.0**	**77.2**

Table 1. Comparing the proposed pixel-level pre-training method, *PixPro*, to previous supervised/unsupervised pre-training methods. For Pascal VOC object detection, a Faster R-CNN (R50-C4) detector is adopted for all methods. For COCO object detection, a Mask R-CNN detector (R50-FPN and R50-C4) with $1\times$ setting is adopted for all methods. For Cityscapes semantic segmentation, an FCN method (R50) is used. Only a pixel-level pretext task is involved in *PixPro* pre-training. For Pascal VOC (R50-C4), COCO (R50-C4) and Cityscapes (R50), a regular backbone network of R50 with output feature map of C5 is adopted for *PixPro* pre-training. For COCO (R50-FPN), an FPN network with P_3-P_6 feature maps is used. Note that InfoMin [30] reports results for only its 200 epoch model, so we reproduce it with longer training lengths, where saturation is observed.

momentum encoder, the momentum value starts from 0.99 and is increased to 1, following [17]. Synchronized batch normalization is enabled during training.

4.2. Downstream Tasks and Settings

We evaluate feature transfer performance on four downstream tasks: object detection on Pascal VOC [15], object detection on COCO [23], semantic segmentation on Cityscapes [10], and semi-supervised object detection on COCO [28]. In some experiments, we also report the ImageNet-1K [11] linear evaluation performance for reference.

Pascal VOC Object Detection We strictly follow the setting introduced in [18], namely a Faster R-CNN detector [27] with the ResNet50-C4 backbone, which uses the *conv4* feature map to produce object proposals and uses the *conv5* stage for proposal classification and regression. In fine-tuning, we synchronize all batch normalization layers and optimize all layers. In testing, we report AP, AP50 and AP75 on the `test2007` set. Detectron2 [33] is used as the code base.

COCO Object Detection and Instance Segmentation We adopt the Mask R-CNN detector with ResNet50-FPN and ResNet50-C4 [19, 22] backbones, respectively. In optimization, we follow the $1\times$ settings, with all batch normalization layers synchronized and all layers fine-tuned [18]. We adopt Detectron2 [33] as the code base for these experiments.

We also consider other detectors with fully convolutional architectures, e.g., FCOS [31]. For these experiments, we follow the $1\times$ settings and utilize the mmdetection code base [6].

Cityscapes Semantic Segmentation We follow the settings of MoCo [18], where an FCN-based structure is used [24]. The FCN network consists of a ResNet-50 backbone with 3×3 convolution layers in the *conv5* stage of dilation 2 and stride 1, followed by two 3×3 convolution layers of 256 channels and dilation 6. The classification is obtained by an additional 1×1 convolutional layer.

Semi-Supervised Object Detection We also examined semi-supervised learning for object detection on COCO. For this, a small fraction (1%-10%) of images randomly sampled from the training set is assigned labels and used in fine-tuning. The results of five random trials are averaged for each method.

ImageNet-1K Linear Evaluation In this task, we fix the pretrained features and only fine-tune one additional linear classification layer, exactly following the settings of MoCo [18]. We report these results for reference.

4.3. Main Transfer Results

Table 1 compares the proposed method to previous state-of-the-art unsupervised pre-training approaches on 4 downstream tasks, which all require dense prediction. Our *PixPro* achieves 60.2 AP, 41.4 / 40.5 mAP and 77.2 mIoU on Pascal VOC object detection (R50-C4), COCO object detection (R50-FPN / R50-C4) and Cityscapes semantic segmentation (R50). It outperforms the previous best unsupervised methods by 2.6 AP on Pascal VOC, 0.8 / 1.0 mAP on COCO and 1.0 mIoU on Cityscapes.

hyp. par.	Pascal VOC AP	AP$_{50}$	AP$_{70}$	COCO mAP	hyp. par.	Pascal VOC AP	AP$_{50}$	AP$_{70}$	COCO mAP
(a) dist. threshold $\mathcal{T}$ using C$_5$					**(d)** layer number in $g(\cdot)$				
$\mathcal{T}$=0.35	58.3	82.1	65.8	39.5	l=0	58.6	82.9	65.4	39.4
$\mathcal{T}$=0.7*	**58.8**	**83.0**	**66.5**	**40.8**	l=1*	58.8	83.0	**66.5**	**40.8**
$\mathcal{T}$=1.4	56.8	82.0	63.3	39.5	l=2	**58.9**	**83.1**	66.3	40.3
$\mathcal{T}$=2.8	56.5	81.7	63.4	39.1	l=3	58.3	82.5	65.0	40.1
(b) dist. threshold $\mathcal{T}$ using P$_3$					**(e)** output resolution				
$\mathcal{T}$=0.35	**58.1**	**83.0**	**64.7**	**40.8**	C$_5$ (7^{2*})	**58.8**	**83.0**	**66.5**	40.8
$\mathcal{T}$=0.7	57.6	**83.0**	63.6	**40.8**	P$_4$ (14^2)	56.7	82.7	63.6	40.9
$\mathcal{T}$=1.4	56.8	82.7	63.1	40.6	P$_3$ (28^2)	57.6	**83.0**	63.6	40.8
$\mathcal{T}$=2.8	56.1	82.4	**64.7**	40.2	P$_3$-P$_6$	55.8	82.5	62.1	**41.3**
(c) sharpness exponent γ					**(f)** training length				
γ=0.5	57.9	82.5	64.5	39.7	50 ep	57.2	82.4	63.4	39.7
γ=1	58.7	83.0	65.5	40.1	100 ep*	58.8	83.0	66.5	40.8
γ=2*	**58.8**	**83.0**	**66.5**	**40.8**	200 ep	59.5	83.5	66.9	40.8
γ=4	58.0	82.4	64.7	40.0	400 ep	**60.2**	**83.8**	**67.7**	**41.0**

Table 2. Ablation studies on hyper-parameters for the proposed *PixPro* method. Rows with * indicate default values.

4.4. Ablation Study

We conduct the ablation study using the Pascal VOC (R50-C4) and COCO object detection (R50-FPN) tasks. In some experiments, the results of the FCOS detector on COCO and semi-supervised results are included.

Hyper-Parameters for *PixPro* Table 2 examines the sensitivity to hyper-parameters of *PixPro*. For the ablation of each hyper-parameter, we fix all other hyper-parameters to the following default values: feature map of C$_5$, distance threshold $\mathcal{T} = 0.7$, sharpness exponent $\gamma = 2$, number of transformation layers in the pixel-to-propagation module $l = 1$, and training length of 100 epochs.

Table 2 (a-b) ablates distance thresholds using the feature maps of C$_5$ and P$_3$. For both, $\mathcal{T} = 0.7$ yields good performance. The results are more stable for P$_3$, thanks to its larger resolution.

Table 2 (c) ablates the sharpness exponent γ, where $\gamma = 2$ shows the best results. A similarity function that is too smooth or too sharp harms transfer performance.

Table 2 (d) ablates the number of transformation layers in $g(\cdot)$, where $l = 1$ shows slightly better performance than others. Note that $l = 0$, which has no learnable parameters in the *pixel-propagation module (PPM)*, also performs reasonably well, while removal of the *PPM* module results in model collapse. The smoothness operation in the *PPM* module introduces asymmetry with respect to the other regular branch, and consequently avoids collapse [17].

Table 2 (e) ablates the choice of feature maps. It can be seen that using the higher resolution feature maps of P$_3$ and P$_4$ performs similarly well to using C$_5$. Using all P$_3$-P$_6$ feature maps noticeably improves the transfer accuracy on COCO object detection, but is inferior to others on Pascal VOC object detection. As the Pascal VOC dataset uses a ResNet-C4 backbone and the COCO dataset uses a ResNet-FPN backbone, this result suggests that consistent architecture between pre-training and the downstream task may give better results.

Table 2 (f) ablates the effects of training length. Increasing the training length generally results in better transfer performance. Our maximal training length is 400. Compared to 200 epoch training, it brings a 0.7 AP gain on Pascal VOC, while almost saturating on COCO. Results on longer training will be examined in our future work.

method	*PPM*	τ	Pascal VOC AP	AP$_{50}$	AP$_{75}$	COCO mAP
PixContrast		0.1	54.7	79.9	61.2	38.0
		0.2	57.1	81.7	63.3	38.6
		0.3	**58.1**	**82.4**	**64.5**	**38.8**
	✓	0.1	52.7	78.8	57.6	37.4
	✓	0.2	53.0	79.1	58.1	37.3
	✓	0.3	52.9	78.8	58.3	37.5
PixPro		-	58.0	82.6	65.6	39.7
	✓	-	**58.8**	**83.0**	**66.5**	**40.8**

Table 3. Comparison of the *PixContrast* and *PixPro* methods. 100 epoch pre-training is adopted for all experiments.

Comparison of *PixPro* and *PixContrast* Table 3 ablates the transfer performance of *PixContrast* with varying τ and with/without the pixel propagation module (*PPM*). It also includes the results of the *PixPro* method with/without the *PPM*. It can be seen that while the *PixContrast* method achieves reasonable transfer performance, the *PixPro* method is better, specifically 0.7 AP and 2.0 mAP better than the *PixContrast* approach on Pascal VOC and COCO, respectively.

Including the pixel-propagation module (*PPM*) leads to inferior performance for the *PixContrast* method, likely because of over-smoothing. In contrast, for *PixPro*, adding *PPM* improves transfer performance by 0.8 AP on Pascal VOC and 1.1 mAP on COCO, as well as avoids the use of hyper-parameter τ. Note while directly removing *PPM* will result in model collapse, we add a linear transformation layer to avoid such collapse issue. Also note that the benefit of this spatial smoothness in representation learning is also evidenced in Table 2(c), where a similarity function that is too smooth or too sharp harms transfer performance.

Combined with Instance-Level Contrastive Methods Table 4 ablates the effects of combining the proposed

PixPro (pixel)	SimCLR* (instance)	VOC AP	COCO mAP	ImageNet top-1 acc
✓		58.8	40.8	55.1
	✓	53.4	40.5	65.4
✓	✓	58.7	40.9	66.3

Table 4. Transfer performance of combining a pixel-level and an instance-level method. "SimCLR*" denotes a variant of SimCLR with the same encoders as our pixel-level approach. 100 epoch pre-training is adopted for all experiments.

+FPN	+head	+instance	COCO (FCOS) mAP	AP_{50}	AP_{75}
			37.8	56.2	40.6
✓			38.1	56.7	41.2
✓	✓		38.6	57.3	41.5
✓	✓	✓	**39.8**	**58.4**	**42.7**

Table 5. FPN and head pre-training with transfer to COCO using an FCOS detector [31]. 100 epoch pre-training is adopted for all experiments.

PixPro method with an instance-level pretext task (SimCLR*) for representation learning. The combination requires marginal added computation due to sharing of the data loader and encoders. It can be seen that an additional instance-level pretext task can significantly improve the linear evaluation accuracy on ImageNet-1K, while the transfer accuracy on COCO (mask R-CNN R50-FPN) and Pascal VOC is maintained. We also observe noticeable transfer improvements of 1.2 mAP on some tasks, e.g., FCOS [31] on COCO, as shown in Table 5.

Effects of Head Network Pre-Training Table 5 ablates head network pre-training (or using an architecture more similar to that in the fine-tuning task) on COCO object detection. For COCO object detection, we use the FCOS detector, which is fully convolutional. We evaluate the transfer performance with an additional FPN architecture, a head network of three successive convolutional layers. It can be seen that more pre-training layers lead to better transfer accuracy on downstream tasks.

Semi-Supervised Object Detection Results Table 6 shows the semi-supervised results using 1% and 10% of labeled data on COCO. The Mask R-CNN (R50-FPN) detector is tested. Our best pre-training models perform significantly better than previous instance-level supervised/unsupervised methods. The gains are +3.9 mAP and +2.3 mAP using 1% and 10% training data, respectively.

The results indicate the advantage of aligning networks between the pre-training and downstream tasks. Including the additional FPN layers in pre-training brings +0.9 and +0.7 mAP gains over the method which pre-trains only the plain backbone network (14.1 and 26.6 vs. 13.2 and 25.9).

arch.	+COCO pre-train	mask R-CNN 1%	10%
supervised		10.4	20.4
MoCo [18]		10.9	23.8
MoCo v2 [9]		10.9	23.9
InfoMin [30]		10.6	24.5
C5 backbone		13.2	25.9
FPN		14.1	26.6
FPN	✓	**14.8**	**26.8**

Table 6. Semi-supervised object detection on COCO. 100-epoch pre-training is adopted for our method, and other methods use the models with their longest training.

We also include an additional pre-training stage on COCO using the proposed pixel-level pretext task by 120 epochs, after the ImageNet-1K pre-training. It leads to additional +0.7 mAP gains and +0.2 mAP gains when 1% and 10% training data are used, respectively. The additional pre-training directly on down-stream unlabelled data may benefit the learning when only scarce labeled data is available.

5. Conclusion

This paper explores the use of pixel-level pretext tasks for learning dense feature representations. We first directly apply contrastive learning at the pixel level, leading to reasonable transfer performance on downstream tasks requiring dense prediction. We additionally propose a *pixel-to-propagation consistency* task which introduces certain smoothness priors in the representation learning process and does not require processing of negative samples. This method, named *PixPro*, achieves 60.2 AP and 41.4 / 40.5 mAP accuracy when the learnt representation is transferred to the downstream tasks of Pascal VOC (Faster R-CNN R50-C4) and COCO object detection (mask R-CNN R50-FPN / R50-C4), which are 2.6 AP and 0.8 / 1.0 mAP better than the previous best supervised/unsupervised pre-training methods. These results demonstrate the strong potential of defining pretext tasks at the pixel level, and suggest a new path forward in unsupervised visual representation learning. As a generic pretext task for learning stronger representations on single images, the proposed approach is also applicable to visual representation learning on videos and multi-modality signals.

References

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views, 2019. 2

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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, 2020. 3

[3] Yue Cao, Zhenda Xie, Bin Liu, Yutong Lin, Zheng Zhang, and Han Hu. Parametric instance classification for unsupervised visual feature learning. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018. 3

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[6] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[7] Mark Chen, Alec Radford, Rewon Child, Jeff Wu, and Heewoo Jun. Generative pretraining from pixels. *Advances in Neural Information Processing Systems*, 2020. 3

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *ICML*, 2020. 1, 2, 4, 5, 6

[9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3, 6, 8

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016. 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. Ieee, 2009. 5, 6

[12] 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. 3

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, pages 1422–1430, 2015. 2

[14] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pages 766–774, 2014. 2

[15] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 6

[16] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 2

[17] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 3, 4, 5, 6, 7

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *CVPR*, 2020. 1, 2, 3, 4, 5, 6, 8

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017. 6

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

[21] Olivier J Hénaff, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *ICML*, 2020. 2, 3

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *ICCV*, pages 2117–2125, 2017. 5, 6

[23] 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. Springer, 2014. 6

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 6

[25] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. *CVPR*, 2020. 1

[26] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 2

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 6

[28] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection, 2020. 6

[29] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 2

[30] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020. 1, 2, 6, 8

[31] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *ICCV*, 2019. 5, 6, 8

[32] Trieu H Trinh, Minh-Thang Luong, and Quoc V Le. Selfie: Self-supervised pretraining for image embedding. *arXiv preprint arXiv:1906.02940*, 2019. 3

[33] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2, 2019. 6

[34] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018. 1, 2

[35] Mang Ye, Xu Zhang, Pong C. Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature, 2019. 2

[36] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 2

[37] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, pages 1058–1067, 2017. 2

[38] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6002–6012, 2019. 2