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

Zhenda Xie*[1,3], Yutong Lin*[2,3], 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

CVPR 2021

Motivation

- Most existing self-supervised learning methods are trained only on instance-level pretext tasks, leading to representations that may be sub-optimal for downstream task requiring dense pixel predictions.

- How to perform self-supervised representation learning at the pixel level is a problem that until now has been relatively unexplored.

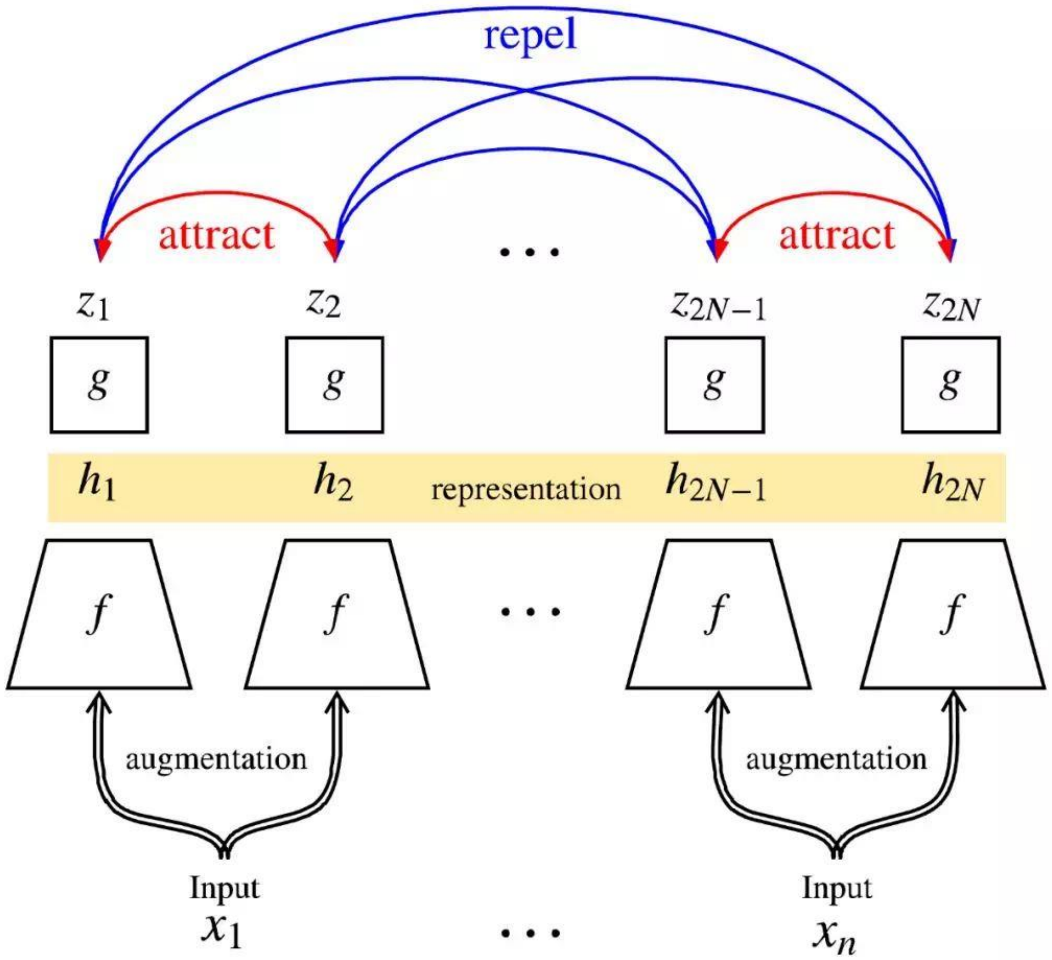
Background

➢ The SimCLR contrastive Learning Framework

- A stochastic data **augmentation** module
- A neural network base **encoder f(.)**
- A small neural network **projection head g(.)**
- A contrastive **loss function L**

➢ Loss function for a positive pair of examples(i,j)

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$



Method

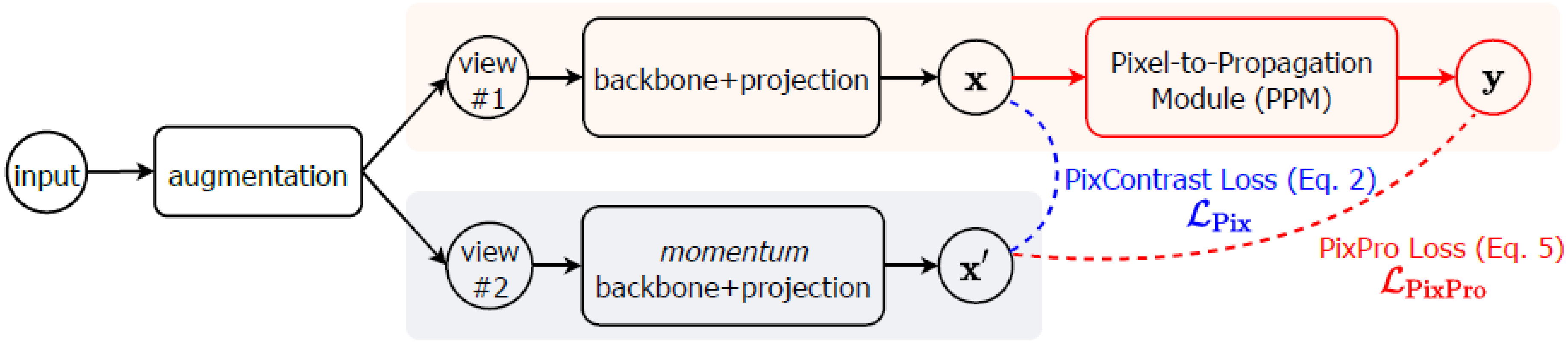


Figure 2. Architecture of the *PixContrast* and *PixPro* methods.

□ Pixel Contrast

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

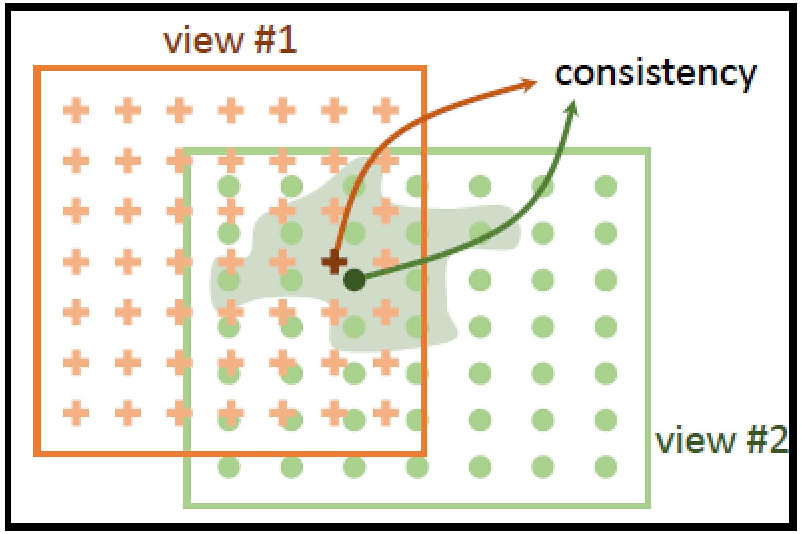
□ PixContrast Loss

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

- **i , j** : pixels form each of two views
- x_i **,** x_j' **:** pixel feature vectors in two views

$\mathbf{\Omega_p^i}$, $\mathbf{\Omega_p^i}$: sets of pixels in the second view assigned as positive and negative with respect to pixel i

PixPro



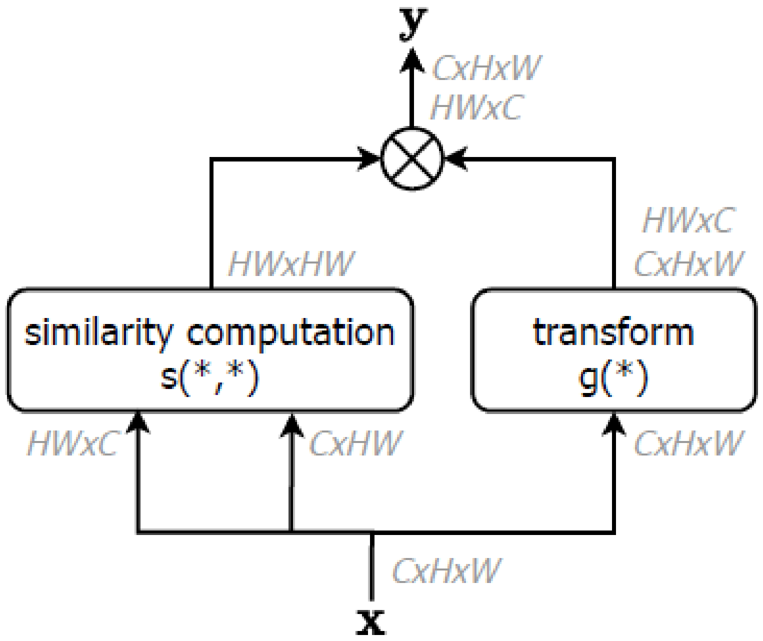


Figure 3. Illustration of the pixel propagation module (*PPM*). The input and output resolutions of each computation block are included.

- Pixel Propagation Module(PPM)

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

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

PixPro

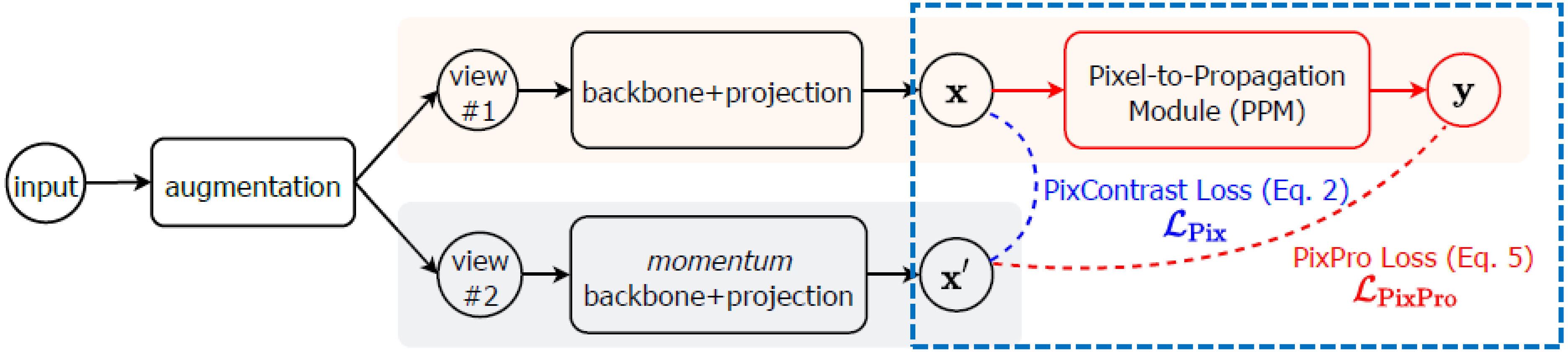


Figure 2. Architecture of the *PixContrast* and *PixPro* methods.

- PixPro Loss

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

It encourages consistency between positive pairs **without consideration of negative pairs.**

Experiment

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 [19]	200	55.9	81.5	62.6	39.4	59.1	43.0	38.5	58.3	41.6	75.3
SimCLR [9]	1000	56.3	81.9	62.5	39.8	59.5	43.6	38.4	58.3	41.6	75.8
MoCo v2 [10]	800	57.6	82.7	64.4	40.4	60.1	44.3	39.5	59.0	42.6	76.2
InfoMin [35]	200	57.6	82.7	64.6	40.6	60.6	44.6	39.0	58.5	42.0	75.6
InfoMin [35]	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× 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 [35] reports results for only its 200 epoch model, so we reproduce it with longer training lengths, where saturation is observed.

Experiment

method	*PPM*	τ	Pascal VOC			COCO
			AP	AP_{50}	AP_{75}	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.

Experiment

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.

Thanks