Dense Siamese Network for Dense Unsupervised Learning: Supplementary Material

Wenwei Zhang[1], Jiangmiao Pang[2], Kai Chen[2,3], and Chen Change Loy[1]✉

[1]S-Lab, Nanyang Technological University
[2]Shanghai AI Laboratory [3]SenseTime Research
{wenwei001, ccloy}@ntu.edu.sg {pangjiangmiao, chenkai}@pjlab.org.cn

A Implementation Details

Data Augmentation. We use the same data augmentation techniques as those used in previous methods for a fair comparison.
i) Unsupervised pre-training: We use existing augmentation modules in PyTorch [6] and describe them using the same notations as the following. Specifically, for geometric augmentation, we use `RandomResizedCrop` with scale in [0.2,1.0] and `RandomHorizontalFlip`. For color augmentations we use `ColorJitter` and `RandomGrayscale` with probabilities of 0.8 and 0.2, respectively. The jittering strength of brightness, contrast, saturation, and hue are 0.4, 0.4, 0.4, and 0.1 in `ColorJitter`, respectively. Blurring augmentation [1] is also applied using a Gaussian kernel with std in [0.1, 2.0]. These hyper-parameters are the same as those adopted in previous methods [1–3, 5, 8].
i) Unsupervised semantic segmentation: We apply the same augmentations as those used in PiCIE [4] in unsupervised semantic segmentation for a fair comparison. The augmentations include color jittering, gray scale, blurring, cropping, and flipping. The color jittering augmentations consists of jittering brightness, contrast, saturation, and hue. These jittering transformations are randomly applied with probabilities of 0.8 with strength of 0.3, 0.3, 0.3, and 0.1, respectively. The gray scale augmentation is randomly applied with a probability of 0.2. Random crop is used with scale in [0.5,1.0]. Different from those augmentations used in unsupervised pre-training, the augmentations are sampled first and replayed with similar parameters in each epoch following PiCIE [4].

B Analysis

Exploitation of Correspondence. Previous methods explore different strategies to build and optimize dense correspondence between two views. DenseCL [8] compares feature similarity to build the correspondence between pixels. PixPro [10] connects pixels by the distance of their coordinates in the original image. There are also attempts [7, 9] to use manual crops and maximize the similarity between similar crops in different views. DenseSiam uses the location and regions for similarity learning of different granularities.

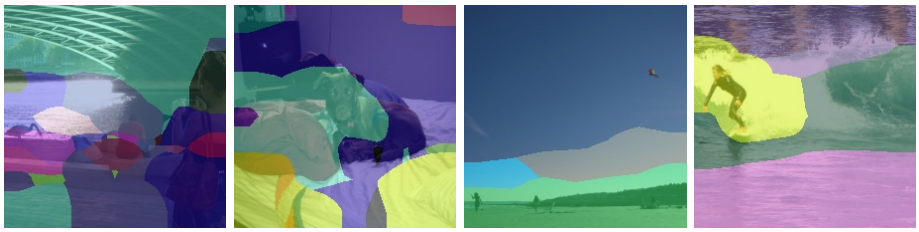
Table A1: **Analysis:** Strategies for building visual correspondence

Strategies	AP	AP_{50}	AP_{75}
SimSiam	53.5	79.7	59.3
feature similarity [8]	36.3	63.5	36.1
pixel distance [10]	53.7	79.3	58.5
candidate regions [7,9]	54.7	79.9	60.5
location	54.9	80.8	60.9
location + region embeddings	**55.5**	**81.1**	**61.5**

Table A2: **Analysis:** Strategies for grid sampling

Strategies	AP	AP_{50}	AP_{75}
uniform (7×7 regular grid)	54.9	80.8	60.9
uniform ($k = 1, \beta = 0.0$)	54.7	80.7	60.3
midly biased ($k = 3, \beta = 0.75$)	55.0	80.7	60.8
heavily biased ($k = 10, \beta = 1.0$)	54.3	80.3	60.0

Fig. A1: **Visualization** of pseudo categories of pixels produced by PixSim in unsupervised representation learning.



For a fair comparison, we study these strategies under the same architecture of PixSim with a switch between strategies. Specifically, we keep the same architecture and symmetrized loss of PixSim and implement these strategies strictly following their official code releases. To study feature similarity used in DenseCL [8], we calculate feature similarities between pixels and link the most similar pixels between views. To study pixel distance used in PixPro [10], we calculate the coordinate distances of pixels and link the pixels having the close locations. To study candidate regions used in ReSim [9], we use anchors in the intersected regions generated by sliding windows.

The results in Table A1 show that using feature similarity [8] significantly decreases the performance. This is because PixSim does not use negative pixel pairs, a prerequisite for the strategy to work. Using pixel coordinates only brings marginal improvements. Using location correspondence in PixSim is much simpler and more effective than candidate regions. DenseSiam yields the best results by exploiting the correspondence built with both location and region embeddings.

Grid Sampling Strategies. We tried a hard example mining strategy proposed in PointRend. Specifically, it first over-generates candidate points by randomly sampling kN points ($k > 1$) from a uniform distribution. Then it estimates the similarities by Eq.3 between the embeddings of these points from both views. Finally, it selects the most dissimilar $\beta N (\beta \in [0, 1])$ points from the kN candidates and sample the remaining $(1 - \beta)N$ points from a uniform distribution.

The results in Table A2 shows that the strategy marginally improves the performance and only training on hard examples degrades the results. More strategies can be explored in future research.

Visualization of masks. When using cross-entropy similarity, $\texttt{softmax}(z'_1)$ can be treated as a segmentation map, in which the pseudo categories of each pixel can be obtained by `argmax` over the channel dimension [3]. We visualize the pixels' pseudo categories by different colors in the figures below. The results show that the pixels are grouped into different pseudo categories without supervision, and the features gathered by these masks thus contain region-level information, which are then forced to have consistency across views in RegionSim.

References

1. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. In: ICML (2020) 1
2. Chen, X., Fan, H., Girshick, R.B., He, K.: Improved baselines with momentum contrastive learning. CoRR **abs/2003.04297** (2020) 1
3. Chen, X., He, K.: Exploring simple siamese representation learning. In: CVPR (2021) 1, 3
4. Cho, J.H., Mall, U., Bala, K., Hariharan, B.: PiCIE: Unsupervised semantic segmentation using invariance and equivariance in clustering. In: CVPR (2021) 1
5. Grill, J., Strub, F., Altché, F., Tallec, C., Richemond, P.H., Buchatskaya, E., Doersch, C., Pires, B.Á., Guo, Z., Azar, M.G., Piot, B., Kavukcuoglu, K., Munos, R., Valko, M.: Bootstrap your own latent - A new approach to self-supervised learning. In: NeurIPS (2020) 1
6. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop (2017) 1
7. Roh, B., Shin, W., Kim, I., Kim, S.: Spatially consistent representation learning. In: CVPR (2021) 1, 2
8. Wang, X., Zhang, R., Shen, C., Kong, T., Li, L.: Dense contrastive learning for self-supervised visual pre-training. In: CVPR (2021) 1, 2
9. Xiao, T., Reed, C.J., Wang, X., Keutzer, K., Darrell, T.: Region similarity representation learning. In: ICCV (2021) 1, 2
10. Xie, Z., Lin, Y., Zhang, Z., Cao, Y., Lin, S., Hu, H.: Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In: CVPR (2021) 1, 2