Towards Efficient and Effective Self-Supervised Learning of Visual Representations

Sravanti Addepalli*, **Kaushal Santosh Bhogale***, **Priyam Dey**, **R.Venkatesh Babu**
Video Analytics Lab, Department of Computational and Data Sciences
Indian Institute of Science, Bangalore, India

Abstract

Self-supervision has emerged as a propitious method for visual representation learning after the recent paradigm shift from handcrafted pretext tasks to instance-similarity based approaches. While the latter have indeed shown promising direction, they require a significantly larger number of training iterations when compared to the supervised counterparts. In this work, we explore reasons for the slow convergence of these methods, and further propose to strengthen them using well-posed auxiliary tasks that converge significantly faster, and are also useful for representation learning. The proposed method utilizes the task of rotation prediction to improve the efficiency of existing state-of-the-art methods. We demonstrate significant gains in performance using the proposed method on multiple datasets.

1 Introduction

Early self-supervised training algorithms [52, 41, 20] aimed at learning representations while solving specialized tasks that require a semantic understanding of the content to accomplish. While generative networks such as task-specific encoder-decoder architectures [30, 53, 45], and GANs [21, 14] could learn useful representations, they were superseded by discriminative tasks such as solving Jigsaw puzzles [41] and rotation prediction [20], as the latter could be achieved using lower model capacities and lesser compute. The surprisingly simple task of rotating every image by a random angle from the set $\{0°, 90°, 180°, 270°\}$, and training the network to predict this angle was seen to outperform other handcrafted task based methods with a similar convergence rate as supervised learning methods [20]. Recent approaches have achieved a significant boost in performance by learning similar representations across various augmentations of a given image [25, 22, 7, 5, 8] at a comparable computational cost. They achieve a further boost when trained for a larger number of epochs [8], indicating that improving their convergence can lead to valuable gains at a low computational cost. We review the related works in greater detail in Section-A1.

In this work, we empirically show that a key reason for the slow convergence of instance-similarity based approaches is the presence of noise in the training objective, owing to the nature of augmentations used, as shown in Fig.1 (Sec.2). We further propose to strengthen the recent state-of-the-art instance-similarity based self-supervised learning algorithms such as BYOL [22] and SwAV [5] using a noise-free auxiliary training objective such as rotation prediction in a multi-task framework [12] (Sec.3). As shown in Fig.2, this leads to a similar convergence rate as RotNet [20], while also resulting in better representations from the instance-similarity based objective. We further study the invariances of the network to geometric transformations, and show in Section-A4.4 that in natural images, rotation invariance hurts performance and learning covariant representations across multiple rotated views leads to improved results. We demonstrate significant gains in performance across CIFAR-10, CIFAR-100 [33], and ImageNet-100 [48, 11] datasets. We further demonstrate the scalability of the proposed approach to a 1000-class dataset using a 30-epoch training schedule on ImageNet-1k (Sec.4, A4).

*Equal contribution.

Correspondence to: Sravanti Addepalli <sravantia@iisc.ac.in>

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

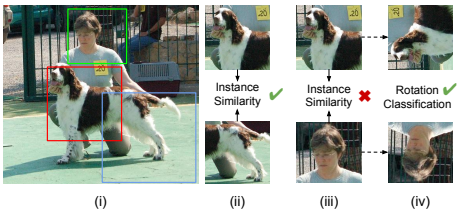


Figure 1: We demonstrate noise in the training objectives of instance-similarity based learning tasks. Consider the three random crops of the input image (i). The two crops in (ii) are desirable, while the crops shown in (iii) give an incorrect signal to the network. In (iv), we show that pretext tasks like rotation prediction can provide a noise-free training objective.

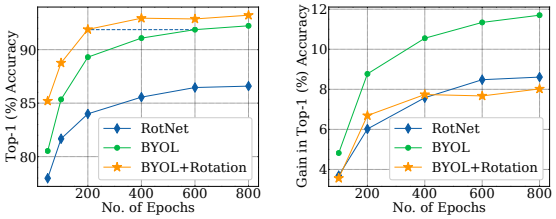


Figure 2: Top-1 Accuracy (%, left plot) after self-supervised pretraining and Linear layer supervised training on CIFAR-10. The proposed method (BYOL+Rotation) achieves the same accuracy as the baseline in one-third the training time (blue dotted line). Gain in Top-1 Accuracy (%, right plot), is the difference between accuracy of the current epoch and epoch-50. The plots show improvements in efficiency and effectiveness of the proposed approach.

Table 1: **Eliminating False Negatives** in contrastive learning across varying levels of supervision (% Labels). Elimination of noise in the training objective leads to higher linear evaluation accuracy (%) within a fixed training budget.

% Labels	SimCLR [7]	Ours	Gain (%)
0	88.77	90.91	2.14
30	92.26 $_{+3.49}$	93.94	1.68
50	92.93 $_{+0.67}$	94.02	1.09
100	93.27 $_{+0.34}$	94.15	0.88

Table 2: **Eliminating False Positives** in BYOL [22] across varying levels of supervision (% Good Crops). Elimination of noise in the training objective leads to higher linear evaluation accuracy (%) within a fixed training budget.

% Good Crops	BYOL [22]	Ours	Gain (%)
0	63.64	68.62	4.98
25	64.50$_{+0.86}$	68.30	3.80
50	66.30$_{+1.80}$	68.90	2.60
100	66.72$_{+0.42}$	70.26	3.54

2 Motivation

In this section, we show that the slow convergence of instance-discriminative algorithms can be attributed to a noisy training objective, and eliminating this noise can lead to improved performance.

Impact of False Negatives in SimCLR: The contrastive learning objective in SimCLR [7] (Eq.A2) considers two augmentations of a given image as positives and the augmentations of all other images in the batch as negatives. These negatives could belong to the same class as the anchor image, and possibly be as similar to the anchor image as the corresponding positive, leading to a noisy training objective. While the probability of same class negatives is higher when batch size is at least $2\times$ higher than the number classes, this issue can occur even otherwise, when there exist negative images that are more similar to the anchor when compared to the positive. Khosla et al. [29] use supervision from labels in a Supervised Contrastive (SupCon) framework to convert the same-class false negatives to additional positives, and show an improvement over supervised learning methods. In order to specifically study the impact of eliminating false-negatives, we perform experiments by using a varying fraction of labels to merely avoid using the same class samples as negatives (without adding these samples as positives), and present results on CIFAR-10 in Table-1. Using 30% labels, we achieve a 3.49% increase in accuracy when compared to the SimCLR baseline (0% labels case).

Impact of False Positives in BYOL: Since BYOL does not use a contrastive learning objective (Eq.A3), it is not directly impacted by noise due to false negatives. However, as shown in Fig.1(iii), the augmentations considered may not be similar to each other, leading to false positives. Selvaraju et al. [47] show that unsupervised saliency maps can be used for the selection of better crops, and also as a supervisory signal in the training objective. Inspired by this, we use Grad-CAM [46] based saliency maps from an ImageNet pre-trained network to select crops for ImageNet-100 pretraining such that the ratio of mean saliency score of the cropped image and that of the full image is higher than a certain threshold and present results in Table-2 (Details in Sec.A2). We observe that by using saliency-maps for crop selection, the accuracy improves by 3.08% for a fixed training budget. While this experiment shows the impact of reducing the false positives in the BYOL objective, it does not completely eliminate noise in the training objective, since the saliency maps themselves are obtained from a Deep Neural Network, and hence may not be very accurate.

The above experiments demonstrate that reduction of noise in the training objective leads to higher accuracy within a fixed training budget, indicating that this can improve the efficiency and effectiveness of instance-discriminative training approaches.

3 Proposed Method

The key ingredients for the success of a self-supervised learning algorithm are (i) Well-posedness of the learning task; (ii) Extent of correlation between representations that help accomplish the pretext task, and ideal representations, whose quality is evaluated using downstream tasks. The success of instance-similarity based approaches in achieving state-of-the-art performance on downstream tasks indeed shows that the representations learnt using such tasks are well correlated with ideal representations. However, these methods require to be trained on a significantly larger number of training epochs when compared to the supervised counterparts. On the other hand, task-based objectives such as rotation prediction score higher on the well-posedness of the learning task. In this task, a known random rotation transformation is applied to an image, and the task of the network is to predict the angle of rotation. Since the rotation angle is known a priori, there is very little scope for noise in labels or in the learning objective, leading to faster training convergence. In this work, we propose to enhance the convergence of instance-similarity based approaches using pretext-task based objectives such as rotation prediction. The proposed approach can be used to enhance many existing instance-discrimination based algorithms (referred to as base algorithm) as shown in Sections-4 and A4. A schematic diagram of our proposed approach is presented in Fig.3.

We term the main feature extractor to be learned as the base encoder, and denote it as f_θ. Some of the self-supervised learning algorithms use an additional encoder, which is derived from the weights of the base encoder. We call this as a derived encoder and represent it using f_ψ. It is to be noted that the derived encoder may be also be identical to the base encoder, which represents an identity mapping between θ and ψ. As proposed by Chen et al. [7], many of the approaches use a learnable nonlinear transformation between the representations and the final instance-discriminative loss. We denote this projection network and its derived network using g_θ and g_ϕ respectively. We note that the base algorithm may have additional layers between the projection network and the final loss, such as the predictor in BYOL [22] and SimSiam [8], which are not explicitly shown in Fig.3.

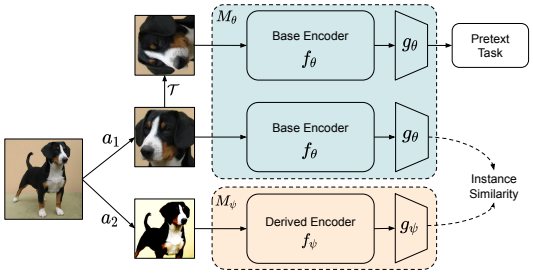


Figure 3: Schematic diagram illustrating the proposed approach. A pretext task such as rotation prediction is combined with base methods like BYOL and SimCLR. For methods like BYOL and MoCo, the derived network M_ψ is a momentum-averaged version of M_θ, and for methods like SimCLR and SimSiam, M_θ and M_ψ share the same parameters.

An input image x is first subject to two augmentations a_1 and a_2 to generate x^{a_1} and x^{a_2}. We use the augmentation pipeline from the respective base algorithm such as BYOL or SimCLR. These augmented images are passed through the base encoder f_θ and the derived encoder f_ϕ respectively, and the outputs of the projection networks g_θ and g_ϕ are used to compute the training objective of the respective base algorithm. The augmentation x^{a_1} is further transformed using a rotation transformation t which is randomly sampled from the set $\mathcal{T} = \{0°, 90°, 180°, 270°\}$. The rotated image $x^{a_1,t}$ is passed through the base encoder f_θ and projection network g_θ which are shared with the instance-based task. We represent the overall network formed by the composition of f_θ and g_θ by M_θ, and similarly the composition of f_ψ and g_ψ by M_ψ. The representation $M_\theta(x^{a_1,t})$ is input to a task-specific network h_θ whose output is a 4-dimensional softmax vector over the outputs in the set $\mathcal{T}$. The overall training objective is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{base}} + \lambda \cdot \frac{1}{2B} \sum_{i=0}^{B-1} \sum_{m=1}^{2} \ell_{CE}(h_\theta(M_\theta(x_i^{a_m,t_k}), t_k)) \tag{1}$$

Here t_k is sampled uniformly at random for each image from the set $\mathcal{T}$. $\mathcal{L}_{\text{base}}$ represents the symmetric loss of the base instance-similarity based algorithm used (Eq.A2, A3). λ is the weighting factor between rotation task and the instance-similarity objective. While the RotNet algorithm [20] uses all four rotations for every image, we consider only two in the overall symmetric loss. Therefore, when compared to the base algorithm, the computational overhead of the proposed method is limited to one additional forward propagation for every augmentation, which is very low when compared to the other components of training such as data loading and backpropagation. There is no additional overhead in backpropagation since the combined loss shown in Eq.1 is used for training.

Table 3: **Transfer Learning (Classification):** Performance (%) after linear evaluation on different datasets with a ResNet-50 backbone trained on ImageNet-1K for 30 epochs.

	ImageNet	CIFAR-10	CIFAR-100	Flowers	Caltech	Aircraft	DTD	Cars	Food	Pets	SUN	VOC	Avg
SwAV [5]	54.90	86.22	64.18	83.53	80.91	38.78	**69.79**	31.65	59.41	70.73	52.48	76.33	64.08
SwAV + Ours	**57.30**	**87.85**	**66.94**	**85.78**	**84.18**	**42.09**	69.68	**32.52**	**59.46**	**71.27**	**53.25**	**76.70**	**65.59**

Table 4: **CIFAR-10, CIFAR-100:** Accuracy (%) of the proposed method when compared to baselines under two evaluation settings - K-Nearest Neighbor (KNN) classification with K=200 and Linear classifier training on CIFAR-10 and CIFAR-100. The proposed method achieves significant performance gains.

Method	CIFAR-10 (200 epochs) KNN	CIFAR-10 (200 epochs) Linear	CIFAR-100 (200 epochs) KNN	CIFAR-100 (200 epochs) Linear
Rotation Pred. [20]	78.01	84.00	36.25	50.87
SimCLR [7]	86.37	88.77	55.10	62.96
SimCLR + Ours	88.69	90.91	57.09	65.40
BYOL [22]	86.56	89.30	54.37	60.67
BYOL + Ours	89.80	91.89	58.41	67.03
SwAV [5]	80.65	83.60	40.35	51.50
SwAV + Ours	85.26	87.20	50.09	58.60
SimSiam [8]	87.05	89.77	56.90	64.27
SimSiam + Ours	**90.35**	**91.91**	**58.92**	**67.38**

Table 5: **ImageNet-100 and ImageNet-1k:** Top-1 and Top-5 Accuracy (%) of the proposed method when compared to baselines under three evaluation settings - Linear classifier training and Semi-Supervised Learning with 1% and 10% labels.

Method	Linear Top-1	Linear Top-5	Semi-Supervised 1% labels Top-1	Semi-Supervised 1% labels Top-5	Semi-Supervised 10% labels Top-1	Semi-Supervised 10% labels Top-5
ImageNet-100 (100 epochs pretraining)						
Rotation Prediction [20]	53.86	81.26	34.72	65.70	51.18	81.38
BYOL [22]	71.02	91.78	46.60	75.50	68.00	89.80
BYOL + Ours	73.60	92.98	56.40	83.50	72.30	91.40
SimCLR [7]	72.02	91.56	57.28	83.69	71.44	91.72
SimCLR + Ours	73.24	92.28	**57.80**	**83.84**	**72.52**	**92.10**
SwAV [5]	72.20	92.96	49.38	78.41	67.56	90.78
SwAV + Ours	**74.40**	**93.33**	52.02	80.01	69.68	91.43
ImageNet-1k (30 epochs pretraining)						
SwAV [5]	49.29	75.01	27.80	52.54	48.76	75.13
SwAV + Ours	**52.57**	**77.72**	**28.43**	**53.90**	**49.73**	**76.02**

4 Experiments and Results

We compare the performance of the proposed method with the respective baselines in the setting of linear evaluation on CIFAR-10, CIFAR-100 (Table-4), ImageNet-100 and ImageNet-1k (Table-5) datasets. We present results on CIFAR-10 and ImageNet-100 datasets with varying number of training epochs in Fig.2 and 7 respectively using BYOL as the base approach. Across all settings, we obtain improved efficiency and effectiveness over the respective baselines. We show transfer learning results with 30 epochs of pretraining on Imagenet-1k (ResNet-50 architecture) in Table-3. We obtain improved results in semi-supervised learning (Table-5) and transfer learning settings as well. We present training details in Sec.A3, details on results in Sec.A4 and ablation experiments in Sec.A5.

Table 6: **Combining BYOL with handcrafted pretext tasks:** Accuracy in (%) after linear evaluation, of various algorithms on ImageNet-100 dataset. Combining instance-similarity based approaches such as BYOL [22] with well-posed tasks such as Rotation Prediction and Jigsaw puzzle solving results in a boost in performance.

	Top-1 Acc (%)	**Top-5 Acc** (%)
RotNet [20]	53.86	81.26
Jigsaw [41]	42.01	72.10
BYOL [22]	71.02	91.78
BYOL + Rotation	73.60	**92.98**
BYOL + Jigsaw	73.60	92.72
BYOL + Rotation + Jigsaw	**74.72**	92.94

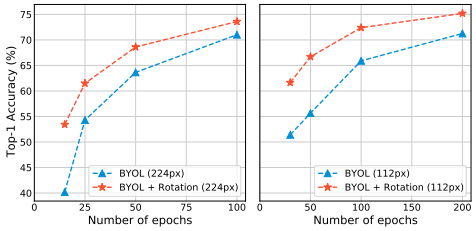


Table 7: Accuracy (%) after Linear layer training for BYOL and the proposed method (BYOL+Rotation) for ImageNet-100. The proposed method achieves significant gains over the baseline in all settings.

5 Conclusions

In this work, we investigate reasons for the slow convergence of recent instance-similarity based methods, and propose to improve the same by jointly training them with well-posed tasks such as rotation prediction. While instance-discriminative approaches learn better representations, handcrafted tasks have the advantage of faster convergence as the training objective is well defined and there is typically no (or very less) noise in the generated pseudo-labels. The complementary nature of the two kinds of tasks makes it suitable to achieve the gains associated with both by combining them, as proposed. Using the proposed approach, we show significant gains in performance under a fixed training budget, along with improvements in training efficiency. We show similar gains in performance by combining the base algorithms with the task of Jigsaw puzzle solving as well (Table-6). We hope that our work will revive research interest in designing specialized tasks, so that they can be help boost the effectiveness and efficiency of state-of-the-art methods.

6 Acknowledgments and Disclosure of Funding

This work was supported by the Qualcomm Innovation Fellowship. We are thankful for the support.

References

[1] Y. M. Asano, C. Rupprecht, and A. Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019. 9

[2] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007. 8

[3] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pages 446–461. Springer, 2014. 14

[4] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018. 9

[5] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 1, 4, 8, 9, 10, 11, 13, 14

[6] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby. Self-supervised gans via auxiliary rotation loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12154–12163, 2019. 8

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 2, 3, 4, 8, 10, 11, 12, 13, 14, 16

[8] X. Chen and K. He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020. 1, 3, 4, 9, 10, 11, 13

[9] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014. 14

[10] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013. 9

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1, 11, 12

[12] C. Doersch and A. Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017. 1, 8

[13] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 8

[14] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *International Conference on Learning Representations*, 2017. 1

[15] L. Ericsson, H. Gouk, and T. M. Hospedales. How well do self-supervised models transfer? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5414–5423, 2021. 14

[16] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 14

[17] R. Faster. Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 9199, 2015. 14

[18] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004. 14

[19] K. Fukushima et al. Self-organizing multilayered neural network. 1975. 8

[20] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 1, 3, 4, 8, 13, 14, 15

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1, 8

[22] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 1, 2, 3, 4, 8, 9, 10, 11, 12, 13, 14, 15, 16

[23] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 11

[24] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 8, 10

[25] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1, 8

[26] O. Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020. 8

[27] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 8

[28] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 10

[29] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020. 2, 12, 14, 16

[30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 8

[31] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2661–2671, 2019. 14

[32] J. Krause, J. Deng, M. Stark, and L. Fei-Fei. Collecting a large-scale dataset of fine-grained cars. 2013. 14

[33] A. Krizhevsky et al. Learning multiple layers of features from tiny images. 2009. 1, 11, 12, 14

[34] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593. Springer, 2016. 8

[35] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017. 8

[36] J. Li, P. Zhou, C. Xiong, R. Socher, and S. C. Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020. 11

[37] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 14

[38] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 14

[39] I. Misra and L. v. d. Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020. 8

[40] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 14

[41] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 1, 4, 8, 14

[42] M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5898–5906, 2017. 8

[43] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 8

[44] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 14

[45] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 1, 8

[46] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 9

[47] R. R. Selvaraju, K. Desai, J. Johnson, and N. Naik. Casting your model: Learning to localize improves self-supervised representations. *arXiv preprint arXiv:2012.04630*, 2020. 2

[48] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 1, 11, 12, 14

[49] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 14

[50] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010. 14

[51] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6721–6729, 2017. 8

[52] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 1, 8

[53] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017. 1

Appendix

A1 Related Works

Early works for learning visual representations from unlabelled images used greedy layer-wise training procedures [19, 27, 2] to find a good initialization for training fully connected architectures. As discussed in Section-1, most of the subsequent methods can be broadly classified into the early handcrafted pretext task based methods, and the more recent instance discrimination based methods.

A1.1 Handcrafted Pretext task based methods

Although pixel-level image generation tasks such as image reconstruction [21, 30] , colorization [52, 34, 35], and inpainting [45, 51] were found to learn useful representations, these tasks required larger capacity models and incurred a higher computational cost for training. Discriminative pretext tasks use pseudo-labels that are generated automatically without the need for human annotations. This includes tasks based on spatial context of images such as context prediction [13], image jigsaw puzzle [41] and counting visual primitives [42].

RotNet: Rotation prediction, proposed by Gidaris et al. [20], has been one of the most successful pretext tasks for the learning of useful semantic representations. In this approach, the network is trained to predict one of the K rotations which was used for transforming the input image x_i. The authors found that $K = 4$ with $\mathcal{T} = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ produced the best results. Every image x_i is transformed using all four rotation transformations $x_i^{t_1}$, $x_i^{t_2}$, $x_i^{t_3}$ and $x_i^{t_4}$, and the network is trained to predict t_1, t_2, t_3 and t_4, which are the rotation angles used for transforming x_i. The base encoder f_θ is trained by minimizing the following loss function $\mathcal{L}$:

$$\mathcal{L}_{RotNet} = \frac{1}{B}\sum_{i=0}^{B-1}\frac{1}{K}\sum_{k=0}^{K-1}\ell_{CE}(M_\theta(x_i^{t_k}), t_k) \tag{A1}$$

Here, M_θ represents the network that takes as input rotated images $x_i^{t_k}$, and outputs the softmax predictions over the four possible rotation angles. Due to its simplicity and effectiveness, the rotation task has been used to improve the training of Generative Adversarial Networks (GANs) [21, 6] as well.

Doersch and Zisserman [12] investigated methods for combining several handcrafted pretext tasks in a multi-task learning framework for learning better representations. Contrary to a general multi-task learning setting, in this work we explicitly consider speeding up instance similarity based tasks such as BYOL [22] and SwAV [5] using handcrafted pretext tasks. We show using controlled experiments that the training objective of instance-similarity based tasks is noisy and the well-defined objective of rotation prediction helps achieve a boost in performance.

A1.2 Instance Discriminative approaches

Recent approaches aim to learn similar representations for different augmentations of the same image, while generating diverse representations across different images. Several works achieve this using contrastive learning approaches [43, 26, 7, 25, 39], where multiple augmentations of a given image are considered as positives, and augmentations of other images are considered as negatives. PIRL [39] and MoCo [25] maintain a queue for sampling a large number of negatives.

SimCLR: The work by Chen et al. [7] presents a Simple Framework for Contrastive Learning of Visual Representations (SimCLR), which utilizes existing architectures such as ResNet [24], and avoids the need for specialized architectures and memory banks. SimCLR proposed the use of multiple data augmentations, and a learnable nonlinear transformation between representations and the contrastive loss to improve the effectiveness of contrastive learning. The authors find the following augmentations to be best suited for the contrastive learning task - random crop and resize, random color jitter and random Gaussian blur. These augmentations are applied serially to every image x_i to generate two independent augmentations $x_i^{a_1}$ and $x_i^{a_2}$, which are considered as positives in the contrastive learning task. The $2(B-1)$ augmentations of all other images in a batch of size B are considered as negatives. The network is trained by minimizing the normalized temperature-scaled cross entropy loss (NT-Xent) loss with temperature T as shown in Eq.(A2). The cosine similarity between two vectors a and b is denoted as $\text{sim}(a, b)$. The overall network formed by the composition of the base encoder f_θ and the projection network g_θ is represented by M_θ.

$$\mathcal{L}_{SimCLR} = -\frac{1}{2B}\sum_{i=0}^{B-1}\sum_{m=1}^{2}\log\frac{\exp(\text{sim}(M_\theta(x_i^{a_1}), M_\theta(x_i^{a_2}))/T)}{\sum_{j=0}^{B-1}\sum_{l=1}^{2}\mathbb{1}_{[j\neq i]}\exp(\text{sim}(M_\theta(x_i^{a_m}), M_\theta(x_j^{a_l}))/T)} \tag{A2}$$

BYOL: While prior approaches relied on the use of negatives for training, Grill et al. [22] proposed Bootstrap Your Own Latent (BYOL), which could achieve state-of-the-art performance without the use of negatives. The two augmentations $x_i^{a_1}$ and $x_i^{a_2}$ are passed through two different networks - the base network M_θ, and the derived network M_ψ respectively. The weights of the base network are updated using back-propagation, while the weights of the derived network are obtained by computing a slow exponential moving average over the weights of the base network. The base network is trained such that the representation of $x_i^{a_1}$ at its output can be used to predict the representation of the $x_i^{a_2}$ at the output of the derived network, using a predictor network P_θ. The symmetric loss that is used for training the base network is shown below:

$$\mathcal{L}_{BYOL} = -\frac{1}{2B}\sum_{i=0}^{B-1}\{\ \text{sim}(P_\theta(M_\theta(x_i^{a_1})), M_\psi(x_i^{a_2})) + \text{sim}(P_\theta(M_\theta(x_i^{a_2})), M_\psi(x_i^{a_1}))\ \} \tag{A3}$$

SimSiam: Chen and He [8] show that it is indeed possible to avoid a collapsed representation even without the momentum encoder using Simple Siamese (SimSiam) networks, and that the stop-gradient operation is crucial for achieving this.

Clustering based methods, SwAV: Clustering-based self-supervised approaches use pseudo-labels from the clustering algorithm to learn representations. DeepCluster [4] alternates between using k-means clustering for producing pseudo-labels, and training the network to predict the same. Asano *et al.*[1] show that degenerate solutions exist in the DeepCluster [4] algorithm. To address this, they cast the pseudo-label assignment problem as an instance of the optimal transport problem and solve it efficiently using a fast variant of the Sinkhorn-Knopp algorithm [10]. SwAV [5] also uses the Sinkhorn-Knoop algorithm for clustering the data while simultaneously enforcing consistency between cluster assignments by Swapping Assignments between Views (SwAV), and using them as targets for training.

A2 Eliminating false positives in self-supervised learning

As shown in Fig.1(iii), two random augmentations of a given image may not always be similar to each other. The use of very small crops increases the likelihood of obtaining augmentations which may be unrelated to each other. This leads to false positives in instance-similarity based learning approaches. In Table-2, we use Grad-CAM [46] based saliency maps to select crops such that mean saliency score of the cropped image is greater than that of the full image. We describe this method in more detail below.

Mean-saliency based cropping: We denote the saliency map of an image using $G(x)$, which is a probability map indicating the importance of each pixel in the image. In order to select rectangular crops having high saliency score, we first calculate the mean probability score $P(x)$ for an image x of dimension $W \times H$ as follows:

$$P(x) = \frac{1}{W \cdot H}\sum_{i=0}^{W}\sum_{j=0}^{H} G_{i,j}(x) \tag{A4}$$

For selecting a rectangular crop from the image, we randomly sample the top left corner coordinates (l, m), width w, and height h from the valid range. These values can be used to obtain a rectangular crop x^{a_1}. We formulate the saliency score of the crop x^{a_1} as follows:

$$P(x^{a_1}) = \frac{1}{w \cdot h}\sum_{i=l}^{l+w}\sum_{j=m}^{m+h} G_{i,j}(x) \tag{A5}$$

The sampled crop is accepted only if $P(x^{a_1}) > P(x)$. We repeatedly sample until a valid crop is found, and restrict to a maximum of 10 tries. If no valid crop is found, we use a random crop. We observe that 10 tries are sufficient to find valid crops in most cases and random cropping is used for very few images.

Computational Budget: As shown in Table-2, with 50 epochs of training, the accuracy on BYOL baseline is 63.64%, which increases to 66.72% with the use of supervised saliency maps. However, this method assumes the availability of a network which is pre-trained on a relevant dataset, which may not always hold true. Hence, the computational budget for training this reference network needs to be considered too. We use fully supervised network trained for 90 epochs as the reference model for generation of saliency maps. Therefore, the total budget for the BYOL baseline is 140 epochs ($50 + 90$). As shown in Table-5, the accuracy obtained by training the BYOL baselines for 100 epochs is 71.02% which is 4.3% higher than the model that is trained for 50 epochs using saliency maps, with an effective training budget of 140 epochs. This shows that while the use of saliency maps from a pre-trained network helps improve accuracy, it is not a practical option in cases where a model that is pre-trained on a related dataset is not available a priori.

A3 Details on Training hyperparameters

We consider the following baselines for our experiments: SimCLR [7], BYOL [22], SimSiam [8] and SwAV [5]. Since these papers primarily demonstrate results on the ImageNet dataset, using larger architectures and longer training schedules, we perform extensive hyperparameter search to obtain strong results for the baselines on the datasets considered. We use the ResNet-18 [24] architecture for all our experiments, unless specified otherwise. The dimension of features before the last fully-connected classification layer is 512, which is smaller than that of ResNet-50, where the dimension is 2048. We fix the batch size to be 512 in all our experiments. We discuss details on hyperparameter tuning for obtaining strong baselines in Section-A3.1, and describe the same for the proposed method in Section-A3.2.

A3.1 Details on the Baseline Implementation

SimCLR: For the SimCLR [7] baseline on CIFAR-10 and CIFAR-100, we perform a hyperparameter search for the learning rate, weight decay and the temperature used in the loss. We tune the learning rate in the range of 0.1 to 1 with a step size of 0.1, and the temperature in the range of 0.1 to 0.5 with a step size of 0.1. For weight decay we search over the range $\{ 5 \times 10^{-4}, 1 \times 10^{-4}, 1 \times 10^{-5}, 1 \times 10^{-6} \}$. Finally, we use a learning rate of 0.5, weight decay of 1×10^{-4} and a temperature of 0.2 for all our experiments. Following the official implementation [7], we use cosine learning rate schedule with a warm-up of 10 epochs. For the projection head, we use a 2 layer MLP with the hidden layer consisting of 512 nodes. The output is a 128-dimensional vector. We use batch normalization layers [28] in the projection head. For ImageNet-100, we use the implementation and tuned hyperparameters from the repository *solo-learn* [2].

BYOL: For BYOL [22] baselines on CIFAR-10 and CIFAR-100, we perform a search for the learning rate and weight decay in the same manner as described in the paragraph above. Additionally we tune the momentum τ of the target network in BYOL [22] from the values $\{ 0.8, 0.85, 0.9, 0.95, 0.99, 0.995, 0.999 \}$. Finally, we use a learning rate of 0.8 and weight decay of 1×10^{-4} for CIFAR-10 and CIFAR-100. We tune the learning rate for ImageNet-100 in the range 0.4 to 0.7 with a step size of 0.1. We finally use a learning rate of 0.6 and a weight decay of 1×10^{-4} for ImageNet-100. We use τ of 0.95, 0.85 and 0.95 for CIFAR-10, CIFAR-100 and ImageNet-100 respectively.

SimSiam: For the SimSiam [8] baselines, we use the implementation from the repository [3], and perform a hyperparameter search for the learning rate, weight decay and the number of projection layers used in the loss. We tune the learning rate in the range of 0.03 to 0.1 with a step size of 0.01, and additionally try 0.2 as well. For weight decay we search over the range $\{ 6 \times 10^{-4}, 5 \times 10^{-4}, 4 \times 10^{-4}, 3 \times 10^{-4}, 1 \times 10^{-4}, 1 \times 10^{-5}, 1 \times 10^{-6} \}$. For the number of projection layers, we consider two values, 2 and 3. Finally, for CIFAR10, we use a learning rate of 0.07, weight decay of 4×10^{-4} and number of projection layers as 2. For CIFAR100, we use a learning rate of 0.05, weight decay of 5×10^{-4} and number of projection layers as 3. Following the official implementation [8], we use the cosine learning rate schedule with a warm-up of 10 epochs. For the projection head, the hidden layer is set to 2048 nodes and output is a 2048-dimensional vector. For the prediction head, the hidden layer has 512 nodes and the output is again a 2048-dimensional vector. We use batch normalization layers in the projection and prediction heads similar to the official implementation [8].

SwAV: We use the code and hyperparameters from the official implementation [5]. For CIFAR-10, we search for the optimal number of prototypes over the values {10, 30, 50, 70, 90, 100, 120, 150}, ϵ over {0.01, 0.03} and queue over {0, 38, 384}. we finally set the number of prototypes to 100 without using a queue, and set ϵ to 0.03. Since CIFAR-10 images are small in size (32x32), we do not use the multi-crop strategy. We use the same settings for CIFAR-100 as well. For ImageNet-100, we scale the default number of prototypes from the official code [5] by a factor of 10 to 300, based on the scaling of number to classes from 1000 to 100. We use search for queue length in the range {0, 384, 1920, 3840} and set it to 384 finally. For the ImageNet-1k runs on ResNet-18, we do not make any changes with respect to the official implementation, whereas for the runs on ResNet-50, we skip the use of multi-crop augmentations to speed up the training.

A3.2 Details on the Proposed Implementation

We use the same hyperparameters as the respective baselines for the implementation of the proposed method, and additionally tune only the value of λ (Eq.1), which is the weighting factor used for the rotation loss. We use a 2 layer MLP for the rotation prediction task and use batch normalization for the hidden layer. For finding the best setting of λ, we tune for $1/(2 \cdot \lambda)$ in the range 1 to 10 with step size of 1, and for $2 \cdot \lambda$ in the range 0 to 1 with a step size of 0.1. In order to minimize computational overheads, we use the same value of λ as ImageNet-100 on ImageNet-1k as well.

[2] https://github.com/vturrisi/solo-learn
[3] https://github.com/PatrickHua/SimSiam

For SimCLR, we use $2 \cdot \lambda$ as 1 for CIFAR-10 and CIFAR-100, and 0.1 for ImageNet-100. For BYOL, we use $1/(2 \cdot \lambda)$ as 5 for CIFAR-10 and CIFAR-100, and 6 for ImageNet-100. For SimSiam, we set the value of $2 \cdot \lambda$ to 0.1 for CIFAR-10 and 0.2 for CIFAR-100. For SwAV, we set the value of $2 \cdot \lambda$ to 0.5 for CIFAR-10 and CIFAR-100, and 0.1 for ImageNet-100 and ImageNet-1k.

A3.3 Training Details of Linear Evaluation

The linear evaluation stage consists of training a linear classification layer on top of the frozen backbone network. We do not update the batch statistics in this stage. For linear evaluation on CIFAR-10 and CIFAR-100, we do not apply any spatial augmentations to the images during training. We use the SGD optimizer with momentum of 0.9. We train for 100 epochs with a batch size of 512. We use a learning rate of 1.0 which is the best setting chosen from the range { 0.1, 0.5, 1.0, 1.5, 2.0 }. The same settings are used for ImageNet-100 BYOL linear evaluation as well.

For SimSiam linear evaluation, we apply Random cropping and horizontal flipping. We use the SGD optimizer with momentum over 100 epochs using a batch size of 512, learning rate of 30.0 and momentum of 0.9, as recommended by the authors [8]. Cosine scheduler with decay is employed without any warmup for the training.

On ImageNet-100, we use the settings from the repository *solo-learn* [4] for the linear evaluation of SimCLR [7]. For linear evaluation of SwAV models on ImageNet-100 and ImageNet-1k, we use the settings from their official repository [5], and use 30 epochs of training on ImageNet-1k.

We use the same hyperparameters for the linear evaluation of the proposed approach and the respective baselines.

A3.4 Training Details of Semi-supervised learning

We follow the semi-supervised training settings from [5, 36] for both 1% and 10% labels. Specifically, we train for 20 epochs with a batch size of 256. For the setting of 1% labels, we use a learning rate of 0.02 for the backbone and 5.0 for the linear layer. For the setting of 10% labels, we use a learning rate of 0.01 for the backbone and 0.2 for the linear layer. We decay the learning rates by a factor of 0.2 at epochs 12 and 16 in both the settings. We do not use weight decay during the training.

A4 Details on Experiments

In this section, we first describe our experimental settings (Sec.A4.1) and details on datasets used for evaluation (Sec.A4.2), following which we present an empirical analysis to highlight the importance of the auxiliary task towards improving the efficiency and effectiveness of the base learning algorithm (Sec.A4.3). We further compare the properties of the learned representations using different training methods and show that learning representations that are covariant to rotation also aids in boosting performance (Sec.A4.4). We compare the results of the proposed method with the state-of-the-art approaches in Sec.A4.5 and show results in a transfer learning setup in Sec.A4.6. We finally show the generality of the proposed method by integrating instance-similarity based methods with other auxiliary tasks such as Jigsaw prediction in Sec.A4.7.

A4.1 Experimental Setup

We run our experiments either on a single 32GB V100 GPU, or across two such GPUs unless specified otherwise. We train all our models on ResNet-18 [23] architecture unless specified otherwise. Our primary evaluations are run for 200 epochs on CIFAR-10 and CIFAR-100 datasets [33], and 100 epochs on ImageNet-100 dataset [48, 11]. We show additional evaluations across varying number of training epochs in Sec.A4.5. We use the respective base algorithm or the proposed approach to learn the base encoder f_θ, and evaluate its effectiveness by training a linear classifier over this, as is common in prior works [7, 8, 22, 5]. In this step, the weights of the base encoder are frozen. We additionally report results in a semi-supervised and transfer learning setting as well. We use the same training hyperparameters as the respective baselines for pre-training, linear evaluation, semi-supervised learning and transfer learning. In the pretraining step, we additionally tune only the value of λ (Eq.1), which is the weighting factor used for the rotation loss.

A4.2 Details on Datasets

We present our analysis and results across the following datasets: CIFAR-10, CIFAR-100 [33] and ImageNet-100 [48], which is a 100-class subset of ImageNet [11]. We do not present our main results on the full ImageNet dataset due to computational limitations. However, we show the scalability of our approach to ImageNet on a short training schedule of 30-epochs. Details of these datasets are presented below:

[4] https://github.com/vturrisi/solo-learn

Table A1: **Task Performance (%):** Evaluation of representations learned using various algorithms on the task of rotation prediction and instance-discrimination.

Method	Linear	Rotation Acc		Contrastive Acc	
		f(.)	g(f(.))	f(.)	g(f(.))
Supervised	94.03	80.54	-	46.36	-
BYOL	89.30	73.40	58.32	78.53	78.82
Rotation	84.00	93.69	93.46	31.61	1.52
BYOL+Rotation	91.89	93.73	93.54	72.85	67.81

Table A2: **Evaluation of BYOL + Rotation** with varying amounts of noise in the rotation task labels. Higher rotation prediction accuracy correlates with higher linear evaluation accuracy.

Rotation Noise	Linear	Rotation Acc		Contrastive Acc	
		f(.)	g(f(.)	f(.)	g(f(.)
30%	89.93	91.88	91.78	73.42	64.25
50%	90.28	89.95	85.82	78.18	77.39
70%	89.75	80.49	67.26	78.55	77.31
80%	89.18	77.43	63.53	77.26	76.92

CIFAR-10: CIFAR-10 [33] is a 10 class dataset comprising of $50,000$ images in the training set and $10,000$ images in the test set. The dataset consists of RGB images of dimension 32×32. The images in the train and test sets are equally distributed across all classes.
CIFAR-100: CIFAR-100 [33] dataset consists of $50,000$ images in the training set and $10,000$ images in the test set, equally distributed across 100 classes. The dimensions and number of channels of images in CIFAR-100 is the same as CIFAR-10.
ImageNet: ImageNet [11] is a 1000-class dataset consisting of around 1.2 million images in the training set and $50,000$ images in the validation set. We consider the validation set as the test set, since the true test set is held private. The dataset consists of RGB images of dimension 224×224.
ImageNet-100: ImageNet-100 is a 100-class subset of the ImageNet dataset. We consider the same 100 class subset that was used by Tian et al. [48].

A4.3 Robustness to Noise in the Training objective

As discussed in Section-2, instance-similarity based tasks such as SimCLR [7] and BYOL [22] suffer from noise in the training objective, and eliminating this noise can lead to significant performance gains in a fixed training budget. We additionally report results of the proposed approach integrated with SimCLR and BYOL in Tables-1 and 2 respectively, and obtain gains over the base approach across varying settings of supervision levels. However, as can be seen from the column Gain (%), the gains using the proposed approach reduce with increasing levels of supervision. This is aligned with our hypothesis that the rotation task helps in overcoming the impact of noise in the base instance-similarity task, and therefore, when additional supervision already achieves this objective, gains using the proposed approach are lower.

Label Noise in a Supervised Learning setting: We consider the task of supervised learning using the supervised contrastive (SupCon) learning objective proposed by Khosla et al. [29]. The training objective is similar to that of SimCLR [7] with the exception that same-class negatives are treated as positives. The authors demonstrate that this method outperforms standard supervised training as well. We choose this training objective as this is similar to the instance-similarity based tasks we consider in this paper, while also having significantly lesser noise due to the elimination of false negatives in training. As shown in Fig.A3a, even in this setting, the proposed method achieves 0.68% improvement, achieving a new state-of-the-art in supervised learning. In order to highlight the impact of noise in training, we run a controlled set of experiments by adding a fixed amount of label noise in each run. The plot in Fig. A3a shows the trend in accuracy of the SupCon algorithm with increasing label noise. The proposed method achieves a significant boost over the SupCon baseline consistently across different noise levels. Further, as the amount of noise in training increases, we achieve higher gains using the proposed approach, indicating that the rotation task is indeed helping overcome noise in the training objective.

We also consider a set of experiments where an equal amount of label noise is added to the SupCon training objective and to the rotation prediction task. We note that in majority of the runs (excluding the case of noise above 70%), the accuracy is very similar to the SupCon baseline with the same amount of noise. This indicates that the knowledge of true labels in handcrafted tasks such as rotation prediction is the key factor that contributes to the improvement achieved using the proposed approach.

We perform the experiments of adding label noise to the rotation prediction task when combined with BYOL and SimCLR as well. As shown in Fig.A3b we find that the gains with the rotation prediction task drops considerably over $0 - 20\%$ label noise, indicating that a similar amount of noise ($\sim 20\%$) is present in the BYOL/ SimCLR training objectives as well. Further, addition of rotation prediction task helps marginally ($0.47 - 1.38\%$) even with higher amount of noise ($30 - 60\%$) in rotation annotations. This indicates that, while the rotation prediction primarily helps by providing a noise-free training objective, it aids the main task in other ways too. We investigate this in the following section.

A4.4 Learning rotation-covariant representations

The task of enforcing similarity across various augmentations of a given image yields representations that are invariant to such transformations. In sharp contrast, the representations learned by humans are covariant with respect to factors such as rotation, color and scale, although we are able to still correlate multiple transformations of the same object very well. This hints at the fact that learning covariant representations could help the accuracy of downstream tasks such as object detection and classification.

In Table-A1, we compare the rotation sensitivity and contrastive task accuracy of representations at the output of the base encoder f_θ, and the projection network g_θ. We follow the process described by Chen et al. [7] to obtain these results. We freeze the network till the respective layer (f_θ or g_θ) and train a rotation task classifier over this using a 2-layer MLP head. We measure the rotation task accuracy, which serves as an indication of the amount of rotation sensitivity in the base network. We further compute the contrastive task accuracy on the representations learned, by checking whether the two augmentations of a given image are more similar to each other when compared to augmentations of other images in the same batch.

Interestingly, a fully supervised network is more sensitive to rotation (80.54%) when compared to the representations learned using BYOL (73.4%). Chen et al. [7] also show that rotation augmentation hurts performance of SimCLR. These observations indicate that invariance to rotation hurts performance, and reducing this lead to better representations. While RotNet has higher accuracy on the rotation task, it does significantly worse on the instance discrimination task, leading to sub-optimal performance compared to BYOL. In the proposed method, we achieve better rotation task accuracy with a small drop in the contrastive task accuracy when compared to BYOL. This also results in an overall higher performance after Linear evaluation.

We also investigate the rotation invariance for the experiments in Sec.A4.3 with BYOL as the base method, where we add noise to the rotation task in the proposed approach. We find that as the amount of noise increases in the rotation task, the amount of rotation invariance increases, leading to a drop in accuracy. Even with 50% noise in the rotation task, we achieve 16.55% boost in rotation performance, leading to 0.98% improvement in the accuracy after linear evaluation. Since the BYOL learning task possibly contains lesser noise compared to this, the gain in performance can be justified by the fact that rotation-covariant representations lead to improved performance on natural image datasets.

A4.5 Comparison with state-of-the-art

We compare the performance of the proposed method with the respective baselines in the setting of linear evaluation on CIFAR-10, CIFAR-100 (Table-4), ImageNet-100 and ImageNet-1k (Table-5) datasets.

We perform extensive hyperparameter search to obtain reliable results on the baseline methods for CIFAR-10 and CIFAR-100, since most existing works report the optimal settings for ImageNet-1k training alone. As shown in Table-4, although the performance of Rotation prediction [20] itself is significantly worse that other methods, we obtain gains of 2.14%, 2.59%, 3.6% and 2.14% on CIFAR-10 and 2.44%, 6.4%, 7.1% and 3.11% on CIFAR-100 when using the proposed method with SimCLR [7], BYOL [22], SwAV [5] and SimSiam [8] respectively. We achieve the best results by combining with BYOL and SimSiam.

We present results on CIFAR-10 dataset with varying number of training epochs in Fig.2 using BYOL as the base approach. Across all settings, we obtain improved results over the BYOL baseline. The proposed method achieves the same accuracy as the baseline in one-third the training time (shown using blue dotted line) as shown in Fig.2. We show the difference in accuracy with respect to accuracy obtained with 50 epochs of training in Fig.2, to clearly visualize the convergence rate of different methods. It can be seen that the proposed method has a similar convergence trend as the Rotation task, while outperforming BYOL in terms of Top-1 Accuracy, highlighting that integrating these methods indeed combines the benefits of both methods.

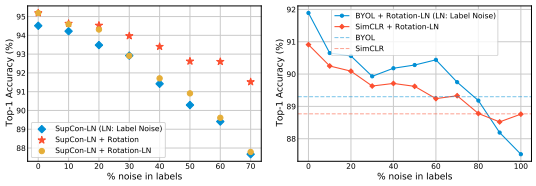
We present results on ImageNet-100 dataset in Table-5. To limit the computational cost on our ImageNet-100 and ImageNet-1k runs, we either use the tuned hyperparameters from the official repository, or follow the settings from other popular repositories that report competent results. Due to the unavailability of tuned hyperparameters on this dataset for SimSiam, we skip reporting results of this method on ImageNet-100. We achieve gains of 2.58%, 1.22% and 2.2% on BYOL [22], SimCLR [7] and SwAV [5] respectively in Top-1 accuracy. We obtain the best results by integrating the proposed method with SwAV, and hence report ImageNet-1k results on the same method, in order to demonstrate the scalability of the proposed method to a large-scale dataset. We restrict our runs to 30-epochs of training due to computational limitations. Using the proposed approach, we obtain a boost of 3.28% in Top-1 accuracy over the SwAV baseline.

We present results on ImageNet-100 dataset with varying number of training epochs in Fig.7. Using the proposed method, we achieve gains across all settings with respect to the number of training epochs.

We obtain improved results over the base methods in semi-supervised learning (Table-5) and transfer learning settings as well.

Table A3: **Transfer Learning (Object Detection):** Performance (AP, AP50 and AP75) on Pascal VOC [16] dataset for the task of Object Detection using Faster RCNN [17] FPN [37] with a ResNet-50 backbone that is pretrained using SwAV [5] and the proposed approach. Pascal VOC07+12 trainval dataset is used for training and VOC07 test is used for evaluation. We consider two settings for evaluation: first with the ResNet-50 backbone being frozen, and second with the backbone being updated during training (Finetune).

Method	VOC (Frozen)			VOC (Finetune)		
	AP	AP50	AP75	AP	AP50	AP75
SwAV [5]	44.10	74.54	45.00	43.80	74.46	45.07
SwAV + Ours	**45.12**	**75.37**	**46.67**	**45.19**	**75.17**	**46.67**



(a) SupCon [29] (b) BYOL, SimCLR

Table A4: The plots demonstrate the impact of label noise in different training objectives on CIFAR-10 dataset. The proposed method (+ Rotation) results in higher performance boost when the amount of label noise in the base method is larger. Addition of label noise to the rotation task reduces the gain in performance.

A4.6 Transfer Learning

In Table-5, we report results on the ImageNet-1k dataset with 30-epoch training on the ResNet-18 architecture, using SwAV [5] as the baseline approach. We use the same hyperparameters as the official implementation for this. In this section, we report results using a ResNet-50 architecture with a 30-epoch training schedule. We perform the pretraining across 4 Nvidia Tesla V100 GPUs. We do not use multi-crop strategy in order to reduce the computational overheads. For all the ImageNet-1k runs, we do not perform additional hyperparameter tuning for the proposed approach, and use the same value of λ that was best in the SwAV ImageNet-100 runs ($2 \cdot \lambda = 0.1$). Using the linear evaluation training code and hyperparameters from the official SwAV repository for 30 epochs on the ImageNet-1k dataset, we achieve 54.9% accuracy using the SwAV baseline, and 57.3% accuracy using the proposed method, resulting in a gain of 2.4% (Table-3). This shows that the proposed approach generalizes well to large-scale datasets and larger model capacities as well.

Classification: We evaluate the generalization of the learned representations to other datasets by training a linear classifier on the pretrained backbone after freezing the weights of the backbone, as reported by Caron et al. [5]. We report transfer learning results on CIFAR-10 [33], CIFAR-100 [33], Oxford 102 Flowers [40], Caltech-101 [18], FGVC Aircraft [38], DTD [9], Stanford Cars [32], Food-101 [3], Oxford-IIIT Pets [44], SUN397 [50] and Pascal VOC2007 [16] datasets, as is common in literature [31, 7, 15]. We use the code, hyperparameter tuning strategy and validation splits from the official repository of Ericsson et al. [15] for obtaining results on the SwAV baseline. For the evaluation of the proposed method, we use the best hyperparameters obtained for baselines, in order to highlight the gains obtained using the proposed approach more clearly. We achieve better performance across most of the datasets, and similar performance as the baseline on the DTD dataset [9]. This is possibly because the DTD dataset is composed of textures only, and the images are rotation invariant. Therefore, learning representations that are covariant to rotation does not help in this case. Overall, we obtain an average improvement of 1.51% across all datasets.

Object Detection: We evaluate the generalization of the learned representations to the task of Object Detection on the Pascal VOC dataset [16] using Faster RCNN [17] with Feature Pyramid Network [37] as the backbone. Pascal VOC07+12 trainval dataset is used for training and VOC07 test is used for evaluation. We consider two settings for evaluation: first with the ResNet-50 backbone being frozen, and second with the backbone being updated during training (Finetune). The training is done using the detectron2 framework [49] and their hyperparameters, as used by Ericsson et al. [15]. As shown in Table-A3, we obtain consistent gains across the metrics AP, AP50 and AP75 in both evaluation settings.

A4.7 Integration with other tasks

In this work, we empirically show that combining instance-discriminative tasks with well-posed handcrafted pretext tasks such as Rotation prediction [20] can indeed lead to more effective and efficient learning of visual representations. While we choose the Rotation prediction task due to its simplicity in implementation, and applicability to low resolution images (such as CIFAR-10), it is indeed possible to achieve gains by using other well-posed tasks as well. We report results on the ImageNet-100 [48] dataset by combining the base BYOL [22] algorithm individually with Rotation prediction [20], Jigsaw puzzle solving [41] and both in Table-6. Although the Jigsaw puzzle solving task is sub-optimal when compared to the Rotation prediction task, we achieve similar gains in performance when these tasks are combined with BYOL. We obtain the best gains (3.7%) when we combine both tasks with BYOL. This shows that the analysis on well-defined tasks being able to aid the learning

Table A5: **Rotation Angles:** Ablation experiments to show the impact of the rotation set ($\mathcal{T}$) used in the proposed approach. K-Nearest Neighbor (KNN) classification accuracy (%) with K=200 and Linear evaluation accuracy (%) on the CIFAR-100 dataset are reported for the baseline (BYOL [22]) and variations in the proposed approach (BYOL + rotation).

Rotation Set ($\mathcal{T}$)	$\lvert\mathcal{T}\rvert$	KNN	Linear
ϕ (BYOL [22])	0	54.37	60.67
$\{0°, 180°\}$	2	58.03	66.21
$\{90°, 270°\}$	2	53.86	62.96
$\{0°, 90°\}$	2	56.41	65.24
$\{0°, 270°\}$	2	56.29	65.04
$\{0°, 90°, 180°, 270°\}$	4	**58.41**	67.03
$\{45°, 135°, 225°, 315°\}$	4	57.60	65.50
$\{0°, 45°, ..., 270° 315°\}$	8	57.54	**67.25**
$\{0°, 30°, ..., 300° 330°\}$	12	55.43	63.61

Table A6: **Effect of number of layers shared with the Rotation Task:** Ablation experiments to show the impact of number of layers shared with the rotation task in the proposed approach. K-Nearest Neighbor (KNN) classification accuracy (%) with K=200 and Linear evaluation accuracy (%) on the CIFAR-100 dataset are reported for the baseline (BYOL [22]) and variations in the proposed approach (BYOL + rotation).

Layers shared with Rotation Task	KNN	Linear
None (BYOL [22] baseline)	54.37	60.67
First Convolutional layer (f_θ)	50.36	52.50
+ Block - 1 (f_θ)	50.98	52.84
+ Block - 2 (f_θ)	51.75	54.85
+ Block - 3 (f_θ)	52.77	58.31
+ Block - 4 (f_θ)	58.29	66.06
+ Projection network (g_θ)	**58.41**	**67.03**

of instance-discriminative tasks that are noisy is indeed generic, and not specific to the Rotation prediction task alone.

A5 Ablation Experiments

In this section, we present additional experiments and results to highlight the significance of various aspects of the proposed method.

A5.1 Impact of Variation in Rotation Angles

In the proposed method, we transform every input image using a rotation transformation $t(.)$ which is randomly sampled from the set $\mathcal{T} = \{0°, 90°, 180°, 270°\}$. We present results by varying the number of rotation angles in the set $\mathcal{T}$ with BYOL [22] as the base approach in Table-A5. While the use of 8 rotation angles results in the best results, we use 4 rotation angles (which results in marginally lower accuracy after linear evaluation) due to the simplicity of implementation, since rotation by multiples of $90°$ does not require additional transformations such as cropping and resizing. The use of two rotation angles with $\mathcal{T} = \{0°, 180°\}$ leads to a drop of 0.82% in linear evaluation accuracy when compared to the proposed method of using 4 rotation angles. However, this setting is still 5.54% better than the BYOL baseline. Therefore, the surprisingly simple task of predicting whether an image is in the correct orientation, or turned upside down is sufficient to boost the performance of the baseline method significantly. In the two-angle prediction task, excluding the $0°$ rotation angle with $\mathcal{T} = \{90°, 270°\}$ leads to a significant drop of 3.25% when compared to using $\mathcal{T} = \{0°, 180°\}$. We further note that using rotation transformations that are uniformly spaced ($\mathcal{T} = \{0°, 180°\}$) leads to better performance when compared to the use of $\mathcal{T} = \{0°, 90°\}$ or $\mathcal{T} = \{0°, 270°\}$.

These experiments show that the level of difficulty of the auxiliary task plays a crucial role in the representations learned. The task should neither be too difficult (12 rotation angles), nor should it be too easy (2 rotation angles). Moreover, since the test images would have $0°$ rotation angle, it helps to include this as one of the classes in $\mathcal{T}$.

A5.2 Impact of Number of Shared Layers across Tasks

In the proposed approach, we share the base encoder f_θ and the Projection network g_θ between the instance-similarity task and the rotation task. We perform experiments to study the impact of varying the number of shared layers between the two tasks. The results of these experiments on the CIFAR-100 dataset with BYOL [22] as the base method are presented in Table A6. The ResNet-18 architecture consists of a convolutional layer followed by 4 residual blocks. As an example, for the case where only Block-1 is shared between the two tasks, we replicate the remaining part of f_θ and g_θ separately for the the rotation task. Thus in this case, the rotation task only impacts Block-1 of the final base encoder f_θ. As shown in Table A6, increasing the number of shared blocks results in better performance. In fact, sharing only the first few layers leads to a degradation in performance when compared to the BYOL baseline. This indicates that the rotation task indeed helps improve the convergence of the overall network, and is not merely helping with learning better filters in the initial layers, as was the case in RotNet [20] training.

Table A7: **Exploring Different Loss Formulations for the Rotation Task:** Ablation experiments to show the impact of different loss formulations on the rotation task. K-Nearest Neighbor (KNN) classification accuracy (%) with K=200 and Linear evaluation accuracy (%) on the CIFAR-10 dataset are reported. We additionally report the Rotation Task Accuracy (%) obtained by freezing the base encoder f_θ and training a 2-layer MLP for the rotation classification task.

	KNN	Linear	Rotation Acc (f_θ)
BYOL Baseline [22]	86.56	89.30	73.40
Ours (Classification with CE Loss)	**89.80**	**91.89**	93.73
Classification with SupCon [29] Loss	88.05	90.19	81.86
Minimizing cosine similarity between Rotation Augmentations	86.84	88.95	77.27
BYOL + Rotation Augmentation	74.32	79.70	66.61
Ours (BYOL + Rotation) + Rotation Augmentation	84.49	87.75	**94.24**

Table A8: **Robustness to Image Augmentations:** Ablation experiments to show the impact of color jitter augmentation on the baseline (BYOL [22]) and proposed method (BYOL + Rotation). K-Nearest Neighbor (KNN) classification accuracy (%) with K=200 and Linear evaluation accuracy (%) on the CIFAR-10 dataset are reported. The proposed method is significantly more robust to the absence of color jitter augmentation.

	KNN	Linear
BYOL [22]	86.56	89.30
BYOL (without Color Jitter)	82.21 −4.35	85.90 −3.40
BYOL + Rotation	89.80	91.89
BYOL + Rotation (without Color Jitter)	88.52 −1.28	91.28 −0.61

A5.3 Robustness to Image Augmentations

BYOL [22] is known to be more robust to image augmentations when compared to contrastive learning methods such as SimCLR [7]. The authors claim that although color histograms are sufficient for the instance-similarity task, BYOL is still able to learn additional semantic features for the image even without color jitter. We compare the impact of removing the color jitter augmentation on the baseline (BYOL) and the proposed approach (BYOL + Rotation) on CIFAR-10 dataset in Table-A8. We observe that addition of rotation task boosts the robustness to such augmentations even further. The absence of color jitter leads to a drop of 3.4% in linear evaluation accuracy of BYOL, whereas the drop in accuracy for the proposed method without color jitter is only 0.61%, which is significantly lower. This makes the proposed method suitable for fine-grained image classification tasks as well, where the network needs to rely on color information for achieving good performance.

A5.4 Exploring Different Loss Formulations for the Rotation Task

The proposed approach combines Cross-Entropy (CE) loss for the rotation task with various instance-similarity based tasks as shown in Eq.1. We explore the use of different loss formulations for the rotation task with BYOL [22] as the base method on the CIFAR-10 dataset in Table-A7. We first replace the CE loss for rotation with SupCon [29] loss, where all images with a similar rotation angle are treated as positives, while the remaining images in the batch are treated as negatives. This results in a significant drop of 1.7% in the Linear evaluation accuracy. We observe a larger drop of 2.94% when the CE loss is replaced with cosine similarity between two unique rotation augmentations sampled from the transformation set $\mathcal{T} = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$. While the three approaches of minimizing CE loss, SupCon loss and cosine similarity between rotation augmentations seek to cluster similarly rotated images together and repel others, we find large differences in the representations learned. This shows that explicitly enforcing fixed categories in the auxiliary task helps in building a global semantic representation which is reinforced across training batches. This is exclusively achieved in the minimization of CE loss since it considers specific rotation based categories.

We study the impact of adding rotations from the set $\mathcal{T} = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ as augmentations in the BYOL training pipeline. Contrary to the proposed approach, this would encourage representations that are invariant to rotation. This leads to a large drop of 9.6% when compared to the BYOL baseline. This is consistent with the observations by Chen et al. [7] that rotation as an augmentation is not helpful in learning good representations. By including the rotation classification task in addition to this in the training objective, the accuracy improves by 8.05%, although it is still lower than the BYOL baseline due to the inclusion of rotation as augmentations which is contrasting to the rotation classification objective.

We further compare the rotation sensitivity of representations at the output of the base encoder f_θ. We freeze the network till the f_θ and train a rotation task classifier over this using a 2-layer MLP head. We measure the rotation task accuracy, which serves as an indication to the rotation sensitivity of the base network. We observe that the trend in accuracy on the linear evaluation task is similar to the rotation task accuracy, indicating that rotation-covariant representations are better for downstream tasks. While the use of rotation augmentation along with rotation task prediction achieves a very high rotation accuracy, its performance on the contrastive task is only 64.34%, which is significantly lower than the baseline and the proposed methods (Table-A1). Therefore, the accuracy on linear evaluation task is also lower than these methods.