Supplemental Material to: An Embedding-Dynamic Approach to Self-supervised Learning

Suhong Moon
UC Berkeley
suhong.moon@berkeley.edu

Domas Buracas
UC Berkeley
dominykas@berkeley.edu

Seunghyun Park
Clova AI Research, NAVER Corp
seung.park@navercorp.com

Jinkyu Kim
Korea University
jinkyukim@korea.ac.kr

John Canny
UC Berkeley
canny@berkeley.edu

Content

This supplementary material provides implementation details (Section 1) including training strategy, architectures, and image augmentations. We also provide evaluation details (Section 2) including linear evaluation protocol, semi-supervised learning setting, k-NN classification, and transfer learning on various downstream tasks. We also report supplemental experimental results of instance segmentation on COCO dataset (Section 3). In addition, this supplementary presents ablation study results. Lastly, we compare our method with [16] and [25] in detail.

1. Implementation Details

We first provide implementation details of our method. We would emphasize that our code will be made publicly available upon publication. In Section 1.1 and , we explain details of our training strategy and architectures. Next, in Section 1.3, we explain details of the stochastic image data augmentation used in our experiment.

1.1. Training Strategy.

We utilize the Layer-wise Adaptive Rate Scaling (LARS) [28] optimizer that is known to effectively overcome large-batch training difficulties. We also use the learning rate scheduler that applies a cosine decay function [19] without restarts to an optimizer step. As suggested by [11], we apply a learning rate warm-up for the first 10 epochs where we start training with a small safe learning rate, which is slowly increased to the max learning rate linearly. The max learning rate is $\texttt{base\_lr} \times \frac{\texttt{batch size}}{256} \times K$ [11]. We set the base learning rate to 0.4 for ImageNet-100, 0.5 for STL-10 dataset and 0.15 for ImageNet datasets. Unless otherwise stated, we set the batch size to 512. The weight decay parameter is set to 1×10^{-5}. We exclude biases and parameters in batch normalization layer following BYOL [13]. We train the model for 320 epochs for ImageNet-100 and STL-10 benchmarks and 300 epochs for ImageNet with 8 V100 16GB GPUs.

1.2. Architectures

For a fair comparison, we use ResNet-18 [15] as a backbone network architecture for STL-10 and ImageNet-100 datasets and ResNet-50 as a backbone for ImageNet dataset, which are widely experimented with conventional approaches for the self-supervised representation learning task. Following BYOL [13], the projection heads (i.e. f_θ and g_ξ in Figure 2 in the main paper) and the prediction head of the online network (i.e. h_θ) use a 2-layer fully connected network with ReLU [20] as an activation function. We tune the size of hidden layers and output layers of projection and prediction heads, when the backbone network is ResNet-18. We use 512 hidden layer size and 128 output layer size instead of 2048 hidden units and 256 output size, which are used in BYOL. We apply batch normalization layer [17]. Also, we experiment various normalization layers including weight standardization [21] and layer normalization [1] to show that our method does not suffer from mode collapse without batch normalization.

1.3. Image Augmentations

We use a stochastic data augmentation operator that is sampled from the family of augmentations $\mathcal{T}$ and results in a randomly augmented view of any given data example. Following SimCLR [4], our data augmentation module sequentially applies the following four augmentations: (1) random cropping followed by resizing back to the original size, (2) aspect-ratio changes, (3) random flipping in the horizontal direction, (4) random color distortion (i.e. jitter and lighting). Detailed augmentation parameters are in Table 1.

Table 1. Image augmentation parameters

Image Augmentation Parameters	Values
1. Random Crop Probability	1.0
2. Flip Probability	0.5
3. Color Jittering Probability	0.8
4. Brightness Adjustment Max Intensity	0.4
5. Contrast Adjustment Max Intensity	0.4
6. Saturation Adjustment Max Intensity	0.2
7. Hue Adjustment Max Intensity	0.1
8. Color Dropping Probability	0.2
9. Gaussian Blurring Probability	1.0
10. Solarization Probability	0.2

2. Evaluation Details

In this section, we provide relevant information for evaluation of our method.

2.1. Linear Evaluation Protocol

We use the linear evaluation protocol [18], which is the standard practice to evaluate the quality of the learned image representations. Using the trained encoder as the feature extractor, we train a linear classifier as a post-hoc manner, i.e. a simple image classifier given a set of features. Then, we measure its classification accuracy on the test set as a proxy of the quality of the learned representations. Note that the encoder is frozen during the evaluation phase. We use the following three standard image classification benchmarks: (1) STL-10 [5], (2) ImageNet-100 [24], and (3) ImageNet [6]. Note that ImageNet-100 contains only 100-class examples that are randomly sampled from ImageNet

2.2. Semi-Supervised Learning

We also evaluate the semi-supervised learning ability of our method with subset of ImageNet training set. We finetune ResNet-50 encoder pretrained with our algorithm and the classifier on top of the encoder using 1% and 10% of ImageNet. These ImageNet splits can be found from the official implementation of [4]. We mainly follow the semi-supervised learning protocol of [13]. We use SGD with momentum of 0.9 and Nesterov, batch size of 1024. We use the separate learning rates for the classifier and the encoder. For fine-tuning task with 1% ImageNet subset, we set learning rate of the classifier 2.0 and freeze the encoder. For fine-tuning task with 10% ImageNet subset, we use the 0.25 as the learning rate of the classifier and 2.5×10^{-4} as the learning rate of the encoder.

2.3. k-NN Classification

We closely follow the existing work [27, 30] to evaluate the quality of representations learned by our model. We first collect representations from training and validation images with the frozen encoder. Then, we compute the classification accuracy of 20/200-nearest neighbor classifier.

2.4. Transferring to Downstream Tasks

To test the transferability of representations trained with our method on ImageNet, we perform transfer learning to various datasets: Places205, iNaturalist2018, Pascal VOC, and COCO.

Image classification. We train the linear classifier layer on top of the frozen ResNet-50 backbone pretrained with MSBReg . For VOC 07, we train a linear support vector machine (SVM). For other image classification benchmarks, iNaturalist 2018 and Places 205, we train the linear classifier with SGD with momentum of 0.9 and weight decay of 10^{-4}. The batch size is 256 and learning rate is 0.2 and we reduce the learning rate by factor of 10 two times with equally spaced intervals. For Places205, the training epoch is 28 and for iNaturalist 2018, the training epoch is 84.

Object detection. Following previous works [12, 3, 2, 10], we finetune the network on VOC07+12 [7] dataset using Faster-RCNN [22]. We report three metrics of the object detection, AP_{all}, AP_{75} and AP_{50}. We use Detectron2 [26] to transfer our model to the object detection task. We set the initial learning rate 0.02. Other hyperparameters such as learning rate scheduling, warm-up steps are exactly same as [14].

Instance Segmentation. For instance segmentation task, we evaluate our model with COCO dataset. We closely follow [14, 29, 2]. We use Mask R-CNN FPN backbone. The backbone is initialized with our pretrained ResNet-50 backbone. We train the network for 90K iterations with a batch size of 16. A learning rate is 0.05 and reduced by a factor of 10 after 60K and 80K iterations. We linearly warm up the learning rate for 50 iterations.

3. Results on COCO Instance Segmentation

We also evaluate the learned representation on COCO insstance segmentation task. We observe in Table 2 that our method shows competitive performance with other methods. Our method is better than BYOL [13] (3rd row), which is our main baseline. SwAV [3] (5th row) shows similar performance to ours. Note that this method uses more augmentations than ours.

4. Related Works

In this section, we supplement Section 2. We compare our work with batch repetition method [16], uniformity loss [25], and BYOL without BN [23] in detail.

Batch Repetition. In the Section 2, we mention batch repetition method [16]. Similar to this method, our multiview

Table 2. Performance comparison for transfer learning on instance segmentation task on COCO dataset. We use `train2017` as training data and report the box detection AP (AP^{bb}) and instance segmentation AP (AP^{mk}) scores on `val2017` dataset.

Method	AP^{bb}	AP^{mk}
SimCLR [9]	39.7	35.8
MoCo [14]	40.4	36.4
BYOL [13]	41.6	37.2
VICReg [2]	39.4	36.4
SwAV [3]	41.6	37.8
BarlowTwins [29]	40.0	36.7
OBoW [10]	40.8	36.4
Ours ($K = 4$)	41.8	37.8

Table 3. Evaluating methods with $\mathcal{L}_{\text{align}}$ and $\mathcal{L}_{\text{uniform}}$

Method	Acc.(%)	Alignment	Uniformity
BYOL	71.9	0.25	-1.52
BYOL+$\mathcal{L}_{\text{uniform}}$	72.1	0.27	-2.95
BYOL+$\mathcal{L}_b + \mathcal{L}_s$	72.8	0.26	-2.92
Ours ($K = 4$)	80.4	0.36	-3.8

centroid loss partially benefits from the fact that simply seeing the same image with different augmentations at each iteration, stabilizes and accelerates training in self-supervised settings. However, the main difference between [16] and multiview centroid loss, is that multiview centroid loss considers the interactions between embeddings of the positive pairs.

Uniformity of Embeddings. In this section, we report uniformity score of MSBReg and other baselines in Table 3. We train BYOL, BYOL with uniformity loss, BYOL+$\mathcal{L}_b + \mathcal{L}_s$ and MSBReg with ImageNet-100 with ResNet-18 backbone. Then, we evaluate each model with three metrics: 1) linear classifier accuracy 2) alignment loss and 3) uniformity loss. Here, both alignment loss and uniformity loss are introduced in [25]. Alignment loss, $\mathcal{L}_{\text{align}}$ is defined as mean squared error between positive pairs and uniformity loss, $\mathcal{L}_{\text{uniform}}$, is defined as the logarithm of the average pairwise Gaussian potential between negative pairs. In Table 3, uniformity loss improves the performance of BYOL (1st row vs 2nd row), by decreasing uniformity loss. Ours shows lower uniformity loss, higher alignment loss and the better performance than other baselines. This strengthen the argument of [25] and ours, which argues that the optimal distribution trained with self-supervised method is uniformly on the embedding manifold.

BYOL without Batch Normalization Layer. The widely known fact about BYOL [13] is that this method falls into

Table 4. Comparison of the quality of representations between BYOL [13] and ours on the STL-10 dataset [5]. The Top-1 classification accuracy is reported with different types of normalization techniques: a batch normalization (BN) [17] and a layer norm (LN) [1]. To see the effect of our proposed Brownian Diffusive Loss, $\mathcal{L}_b$, we also report scores of BYOL with $\mathcal{L}_b$ (4th row).

Method	Norm. Layer	Batch Size	λ_b	Top-1 (%)
BYOL	BN	256	0	89.5
Ours	BN	256	5×10^{-2}	91.4
BYOL	LN	256	0	10.6
BYOL + our $\mathcal{L}_b$	LN	256	5×10^{-3}	75.3
BYOL	LN	1024	0	10.6
Ours	LN	256	5×10^{-4}	80.7
Ours	LN	256	5×10^{-3}	82.3
Ours	LN	256	5×10^{-2}	78.7

the mode collapse [8] without batch normalization layer. The authors of [13] performed studies that BYOL works even without BN layer [23]. In this paper, authors showed that BYOL without BN gets matched performance using various normalization techniques including weight standardization [21] or the deliberately handled initialization. But still, BYOL fails to converge optimal solution with such deliberately tuned training techniques. In this section, we show that MSBReg also work with layer normalization [1] without any other techniques in Table 4.

In Table 4, the top-1 classification accuracy is largely degraded from 89.5% to 10.6%, i.e. mode collapsed. Ours with the Brownian diffusive loss $\mathcal{L}_b$ was not the case (compare 2nd vs. 6th row). Though we observe a slight degradation in the top-1 classification accuracy, ours sufficiently avoid collapsed representations. Further, we evaluate the BYOL with our Brownian diffusive loss to demonstrate its effectiveness against a mode collapse. We observe that our Brownian diffusive loss helps avoid collapsed representations (compare 3rd vs. 4th rows). We also observe that the quality of representations depends on the strength of the hyperparameter λ_b where we obtain the best performance with $\lambda_b = 5 \times 10^{-4}$. We observe a tension as we see a smaller or larger λ_b slightly degrades the quality of representations.

5. Ablation Studies

We perform ablation experiments to study the trade off between major hyperparameters in MSBReg , λ_s and λ_b. In table 5, our experiment reports the top-1 classification accuracy on ImageNet-100. We train ResNet-18 with MSBReg for 300 epochs with various combinations of $K \in \{2, 4, 8\}$, $\lambda_s \in \{0, 0.002, 0.004, 0.01\}$, and $\lambda_b \in \{0, 0.25, 0.5, 1.0, 2.0\}$. Note that the case of $K = 2$ is the same as BYOL setting. Then, we train the linear classifier on top of frozen ResNet-18 backbone pretrained with MSBReg . Our study shows that the classification accuracy increases until λ_s=0.004, $\lambda_b = 0.5$ for the cases of $K = 4$ and $K = 8$. Interestingly, both singular value loss and Brownian loss improve the performance for the case of BYOL ($K = 2$).

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. 2020.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[5] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. JMLR Workshop and Conference Proceedings.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015.

[8] Abe Fetterman and Josh Albrecht. Understanding self-supervised and contrastive learning with bootstrap your own latent (byol).

[9] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick P'erez, and Matthieu Cord. Learning representations by predicting bags of visual words. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6926–6936, 2020.

[10] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. Obow: Online bag-of-visual-words generation for self-supervised learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6826–6836, 2021.

[11] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[12] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[16] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[18] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning, 2019.

[19] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

[20] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[21] Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Micro-batch training with batch-channel normalization and weight standardization, 2020.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[23] Pierre H. Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, and Michal Valko. Byol works even without batch statistics, 2020.

[24] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding, 2020.

[25] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere, 2020.

[26] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[27] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference*

Table 5. Ablation studies to investigate the trade-off between losses in MSBReg .

K	λ_s	λ_b	Acc. (%)	K	λ_s	λ_b	Acc. (%)	K	λ_s	λ_b	Acc. (%)
2	0	0	71.9	4	0	0	78.2	8	0	0	79.5
2	0	0.25	72.3	4	0	0.25	79.1	8	0	0.25	80.1
2	0	0.5	72.8	4	0	0.5	79.7	8	0	0.5	80.3
2	0	1.0	72.3	4	0	1.0	79.2	8	0	1.0	80.3
2	0	2.0	71.9	4	0	2.0	79.0	8	0	2.0	80.2
2	0.002	0	72.2	4	0.002	0	78.9	8	0.002	0	79.8
2	0.002	0.25	72.4	4	0.002	0.25	78.8	8	0.002	0.25	80.2
2	0.002	0.5	72.4	4	0.002	0.5	79.1	8	0.002	0.5	80.9
2	0.002	1.0	72.1	4	0.002	1.0	79.2	8	0.002	1.0	81.1
2	0.002	2.0	71.9	4	0.002	2.0	79.2	8	0.002	2.0	80.8
2	0.004	0	72.8	4	0.004	0	79.7	8	0.004	0	80.0
2	0.004	0.25	72.8	4	0.004	0.25	80.2	8	0.004	0.25	80.9
2	0.004	0.5	72.4	4	0.004	0.5	80.4	8	0.004	0.5	81.6
2	0.004	1.0	72.1	4	0.004	1.0	80.1	8	0.004	1.0	81.5
2	0.004	2.0	71.2	4	0.004	2.0	79.9	8	0.004	2.0	81.3
2	0.01	0	71.1	4	0.01	0	79.3	8	0.01	0	79.9
2	0.01	0.25	71.0	4	0.01	0.25	79.4	8	0.01	0.25	81.2
2	0.01	0.5	71.0	4	0.01	0.5	79.2	8	0.01	0.5	81.4
2	0.01	1.0	70.7	4	0.01	1.0	79.2	8	0.01	1.0	81.1
2	0.01	2.0	70.3	4	0.01	2.0	79.0	8	0.01	2.0	79.8

on Computer Vision and Pattern Recognition (CVPR), June 2018.

[28] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[29] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction, 2021.

[30] Chengxu Zhuang, Alex Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6001–6011, 2019.