Training Barlow Twins with Small Batch Sizes by Using a Queue of Previous Outputs

Pedro de Carvalho Cayres Pinto and José Gabriel Rodríguez Carneiro Gomes

***Abstract*— We present two methods based on Barlow Twins, a self-supervised method, to improve training with smaller batches. The first method randomly drops features from the output before computing the loss to reduce the variance. The second method introduces a queue of outputs from previous batches to improve the loss estimate during training. The first method, with a batch size of 64, achieves an accuracy of 64.1%, the second method, with a batch size of 64 and 192 queued outputs, achieves an accuracy of 65.4%, while the original method, with a batch size of 256, achieves an accuracy of 66.0%.**

***Keywords*— self-supervised learning, convolutional neural networks, deep learning**

I. Introduction

Recent methods in self-supervised learning, such as [1]–[7], have shown similar performance to supervised methods in image classification tasks on ImageNet [8]. When comparing their Top-1 accuracy on the ImageNet validation set with a ResNet-50 [9] architecture, Barlow Twins [1] has an accuracy of 73.2%, SwAV [5] has an accuracy of 71.8% (75.3% with multi-crop), BYOL [6] has an accuracy of 74.3%, and VICReg [7] has an accuracy of 73.2%, which are close to the supervised accuracy of 76.5%.

For object detection and image segmentation, models pre-trained on ImageNet with self-supervised techniques outperform the ones pre-trained with supervision. According to [1], when using these pre-trained models as the base of the Mask R-CNN [10] architecture, the bounding box average precision (AP) is 39.3% with MoCo-v2 [4], 38.4% with SwAV, and 39.2% with Barlow Twins, against 38.2% with a pre-trained supervised network. Similarly, according to [1], in the segmentation task, MoCo-v2, SwAV, and Barlow Twins have a mask AP of 34.4%, 33.8%, and 34.3%, respectively, while a pre-trained supervised network achieves 33.3% mask AP [1].

Despite these advancements, to achieve such a performance the training is costly and requires more than 100 hours on 32 V100 GPUs to train 1000 epochs [1]. With a batch size of 256 these models can be trained with only four V100 16 GB GPUs but we show that, with some modifications, we can reduce the batch size to 64 and obtain similar performance training only with two 1080 Ti 11 GB GPUs.

Pedro de Carvalho Cayres Pinto, Electrical Engineering Department - COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro-RJ, e-mail: pedrocayres@coppe.ufrj.br; José Gabriel Rodríguez Carneiro Gomes, Electrical Engineering Department - COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro-RJ, e-mail: gabriel@pads.ufrj.br. This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq/Brazil), grant numbers 142264/2019-9, 440074/2020-7 and 309089/2022-0, and in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) Finance Code 001.

The loss functions of these self-supervised learning methods are often based on the idea that the output features from similar images should also be similar, with a pair of images being considered similar when they are both produced by a pair of randomly sampled transformations applied to the same image in a data augmentation process. Pairs of such similar images, as well as their outputs produced by the network, are referred to as the *positive pairs*.

From a batch of different images sampled from the dataset, the augmentation process generates two matching batches so that each image from the original batch has a transformed version on each batch. Each of these batches are processed by a network (possibly with the same weights, as in the case with Barlow Twins), generating two batches of output feature vectors.

In the contrastive methods, such as [2]–[4], the output feature vectors are compared directly by cosine similarity, and the cross-entropy loss, calculated for each sample, is smaller when the positive pairs have higher cosine similarity and the negative pairs have lower cosine similarity.

Unlike the contrastive methods, Barlow Twins [1] does not compare the negative output pairs directly. Instead, this method computes an estimate of the cross-correlation matrix between the output vectors of the two networks and the loss is the mean squared error between this cross-correlation matrix and the identity matrix. With this loss, networks produce output vectors where pairs of features are highly correlated when they have the same index, and are from a positive output pair, and where the pair of features are uncorrelated otherwise.

In this work, we introduce modifications on the Barlow Twins method in order to alleviate the performance reduction that is caused by training with small batch sizes. These modifications aim at improving the optimization process by reducing the loss estimate variance. We test these modified Barlow Twins methods with a batch size of 64 on the ImageNet dataset [8] and achieve performance similar to the performance of the original Barlow Twins method with a batch size of 256.

While our work focuses on training convolutional neural network with self-supervision, the advent of vision transformers (ViT) [11] led to the development of self-supervised techniques such as masked autoencoders (MAE) [12] and self-distillation with no labels (DINO [13], DINOv2 [14]).

In Section II we review the Barlow Twins loss and propose changes to improve the loss estimation with batches. The implementation of these changes is detailed in Section III. We apply this methodology by performing experiments on Imagenet and CIFAR-10 [15] and discuss the results of these experiments in Section IV. Finally, Section V presents our conclusions and directions for future work.

II. Background

In the original Barlow Twins method, two twin networks are trained (in the implementation, it is a single network, as explained in the second paragraph of Section IV). At first, a batch of images is sampled from the dataset. Then, for each image in the batch, two transformations are sampled from the augmentation process and applied to the corresponding image, generating two versions of the batch. Each twin network processes one of these batches and yields an output feature vector for each image. A cross-correlation matrix between these output vectors is computed. Finally, the loss is defined as the mean squared difference between the cross-correlation matrix and the identity matrix (with a hyperparameter λ multiplying the off-diagonal error component). The idea is that the loss is smaller when the output vectors from two different transformations are similar (the diagonal elements of the matrix are close to 1) and when the correlation between features of different index from each output vector pair is low.

More specifically, if x is a batch of images from the dataset, each of these images is transformed twice by the data augmentation process, thus generating two batches of transformed images, x^A and x^B. Each batch is processed by the network f_θ with parameters θ, thus producing the output vector batches $y^A = f_\theta(x^A)$ and $y^B = f_\theta(x^B)$. These vectors are normalized by subtracting from the mean and dividing by the standard deviation along the image indices within the batch (as is done in batch normalization). From these output vectors, the correlation coefficients, $\hat{C}_{ij}$, are calculated as in (1), where the first subscript of y is the index of the image within the batch and the second subscript of y is the index of the feature within the vector. Finally, the loss is calculated from the cross-correlation matrix as in (2).

$$\hat{C}_{ij} = \frac{\sum_k y^A_{k,i} y^B_{k,j}}{\sqrt{\sum_k {y^A_{k,i}}^2}\sqrt{\sum_k {y^B_{k,i}}^2}} \tag{1}$$

$$\hat{\mathcal{L}}_{\text{BT}} = \sum_i (1-\hat{C}_{ii})^2 + \lambda \sum_{i,j,i\neq j} \hat{C}_{ij}^2 \tag{2}$$

Loss functions can often be computed directly as a function of the output, possibly taking into consideration a target as well. In such cases, usually, the true loss is the expected value of the loss function over the generating process (in practice, the dataset), while the estimated loss is the average loss function computed over each output of a batch of samples. If the batch $\{x_i\}_{i=1}^n$ is sampled from the data distribution $\mathcal{D}$, the fact that this loss estimate $\hat{\mathcal{L}}$ is unbiased naturally follows from the linearity of the expected value, as explained by (3).

$$\begin{gathered}\mathop{\mathbb{E}}_{\{x_i\}_{i=1}^n \sim \mathcal{D}}[\hat{\mathcal{L}}] = \mathop{\mathbb{E}}_{\{x_i\}_{i=1}^n \sim \mathcal{D}}\left[\frac{\sum_{i=1}^n l(f_\theta(x_i))}{n}\right] = \\ \sum_{i=1}^n \frac{\mathop{\mathbb{E}}_{x_i\sim\mathcal{D}}[l(f_\theta(x_i))]}{n} = \mathop{\mathbb{E}}_{x\sim\mathcal{D}}[l(f_\theta(x))] = \mathcal{L}\end{gathered} \tag{3}$$

However, in Barlow Twins, due to the bias and variance of the estimate of the correlation coefficients, the bias of the estimated loss exists and is dependent on the batch size. In fact, if C_{ij} is the true correlation coefficient of an off-diagonal element, we have (4) (this is just the well-known equation, $\mathbb{E}[X^2] = \mathbb{E}[(X-\mathbb{E}[X])^2] + \mathbb{E}[X]^2$, applied to $\hat{C}_{ij}$), where $\sigma^2_{\hat{C}_{ij}}$ is the variance of $\hat{C}_{ij}$, and $b_{\hat{C}_{ij}} = \mathbb{E}[\hat{C}_{ij}] - C_{ij}$ is the bias of $\hat{C}_{ij}$, and the expected value is taken over the distribution of samples from the image generating process, and the distribution of transformation pairs for each sample.

$$\mathbb{E}[\hat{C}_{ij}^2] = \sigma^2_{\hat{C}_{ij}} + (C_{ij} + b_{\hat{C}_{ij}})^2 \tag{4}$$

If, in a hypothetically ideal case, the output vectors from the network, before normalization, follow a multivariate i.i.d. normal distribution and the network generates the same output vector for each sample image for every possible transformation, then the true cross-correlation matrix is the identity matrix and the true loss is indeed 0, but the expected value of the estimated loss is $\lambda\frac{d(d-1)}{n-1}$, for a batch size of n and an output vector with dimension d. Each off-diagonal element accounts for $\frac{\lambda}{n-1}$ and the diagonal elements are always 0. For the interested reader, derivation of the distribution, bias and moments of the correlation coefficient in the bivariate normal distribution case is provided in [16].

In our experiments, training with batch sizes smaller than 128 diverged in the early epochs, which makes the original method impractical to execute with constrained hardware resources. We consider the hypothesis that the divergence is caused by the noise of the loss estimation process and propose two modified versions of Barlow Twins that make the method viable with a batch size of 64.

III. Methodology

The first modified method drops out some of the features of the output vectors before computing the loss. We remove each feature with a 50% probability. This process reduces the cross-correlation matrix to around 1/4 of its original size and simplifies the loss for each batch. This should have an effect on the magnitude and variance of the loss similar to increasing the batch size from 64 to 256. While this modification does not reduce the bias of each correlation coefficient, it does reduce the variance of the loss, and is enough to lead to convergence in our experiments. This idea can also be used to explore higher dimensionality of the output vectors at a reduced hardware cost. As observed in the Barlow Twins paper [1], increasing the dimensionality of the outputs, at least up to 16384, improves the accuracy of the finetuned model.

In the current implementation, we simply change the feature at the selected index to zero, but a more efficient implementation should also reduce the output size, in order to reduce the memory and time to compute the cross-correlation matrix.

The second modified method implements a queue, from outputs from previous batches, and uses the queue and the outputs from the current batch to compute a more precise estimate of the cross-correlation matrix. This follows a similar idea to the one in MoCo [3], [4], where a queue of previous outputs is used in conjunction with the batch outputs to compute the contrastive loss, and in SwAV [5] in the small batch setting, where the queue augments the batch outputs to compute the assignments to the prototypes.

The output vectors, before batch normalization, of each batch are pushed into the queue after they are calculated, discarding the oldest output vectors in the queue. This is done for both twin networks, so there are two queues. Each queue is appended to the output vectors generated by one network in the pair, before the batch normalization operation, to compute the cross-correlation matrix. We use a queue size of 192 and a batch size of 64, which simulates a batch size of 256 when computing the cross-correlation matrix. The queue is initialized with an i.i.d. normal distribution with mean 0 and standard deviation 1.

For clarity, it is useful to consider the trained networks as being divided into two smaller networks. The first one is the base network, such as a ResNet without the classification layer, that produces the output features that are called the *representations*. The second one is the projector network which is composed, in this case, of three fully connected layers that process the representations of the first network and produce the output vectors. The first two fully connected layers are followed by batch normalization [17] and ReLU, while the output of the last layer is used to compute the cross-correlation matrix. The whole architecture is randomly initialized.

As noted in [2], removing the projector network after finishing the self-supervised training produces better results in the classification task.

IV. Results

We perform one experiment to compare three methods. The first method is the baseline Barlow Twins with a batch size of 256. The second method uses a batch size of 64 and modifies baseline by sampling (with a 50% chance) indices of the output vectors to exclude from the loss computation at each training step. The third method uses a batch size of 64 and uses a queue of size 192 with outputs from previous batches, that is concatenated to the current output batch in order to compute a more precise loss. The 192 outputs in the queue complement the 64 outputs in the batch to simulate an overall batch size of 256.

In the 64 batch size settings, the models are trained with two 1080 Ti 11 GB GPUs of a single machine. To train the model with a batch size of 256 we use all four V100 16 GB GPUs of a cloud instance. This training with 256 batch size could not be realized in a single machine with four 1080 Ti GPUs due to the memory requirement.

The base convolutional network architecture is a ResNet-50 [9] and the dimensions of the three fully connected layers in the projector network are 2048×8192, 8192×8192 and 8192×8192. The twin networks share their weights. That is to say, they are implemented by a single network, which processes both batches in the pair.

The image augmentation settings are the same as those used in Barlow Twins [1] and BYOL [6]. The image is randomly cropped. After that, with random chance, horizontal flipping, color jittering, solarization and Gaussian blur are applied. The images are resized to 224×224 and normalized along each channel by subtracting the mean and dividing by the standard deviation from the ImageNet train set.

TABLE I: Accuracy results on the ImageNet validation set after finetuning the classification layer. The batch size is B and the queue size is Q.

Method	Top-1	Top-5
Barlow Twins ($B = 256$)	66.0%	86.8%
Barlow Twins ($B = 64$) with drop feature	64.1%	85.8%
Barlow Twins ($B = 64$) with queue ($Q = 192$)	65.4%	86.8%

The optimization algorithm is layer-wise adaptive rate scaling (LARS) [18] on the weights and stochastic gradient descent (SGD) without weight decay on the biases and batch normalization parameters, following [1]. In the baseline method, the base weight learning rate[1] is 0.4 and the base bias learning rate is 0.0096, while on the method that drops features and the queue method, the base weight learning rate is 1.0 and the base bias learning rate is 0.003. In all methods the weight decay is 10^{-6}, the momentum is 0.9 and $\lambda = 0.0051$, following [1].

The self-supervised models train for 100 epochs on the ImageNet train set, with a 10-epoch warm-up period, which raises the learning rate linearly to its base value. After warm-up, the learning rate follows a cosine decay learning rate schedule that finishes with a learning rate of $\frac{1}{1000}$ of the base learning rate.

For classification, we include a 2048×1000 fully connected layer on top of the representations of the pre-trained self-supervised model (the projector network is removed), so that we have one output for each one of the 1000 ImageNet classes. We train this fully connected layer with the cross-entropy loss on the ImageNet train set for 100 epochs with SGD, learning rate of 0.3, weight decay of 10^{-6}, momentum of 0.9 and cosine decay schedule.

The Top-1 and Top-5 accuracy results on the ImageNet validation set are shown on Table I. The baseline method yields the best performance with an accuracy of 66.0% and a Top-5 accuracy of 86.8%. The method with a queue shows similar performance with an accuracy of 65.4% and a Top-5 accuracy of 86.8%. The method that drops features from the outputs presents a reduction in accuracy in both cases, Top-1 and Top-5, with an accuracy of 64.1% and 85.8% respectively, in comparison with the baseline Barlow Twins. The evolution of the accuracy for each method, by epoch, is shown in Figure 1 and Figure 2.

The Barlow Twins loss is shown in Figure 3. At the final epoch there is a gap of around 7.5% of the loss in the baseline method between the baseline method with a batch size of 256 and the queue method with a simulated batch size of 256 (64 batch size plus 192 from the queue). Despite the gap, the accuracy performance of the queue method is similar to the

[1]These learning rates are reported as in the Barlow Twins paper [1], but we note that, like in the original code available at `https://github.com/facebookresearch/barlowtwins`, the actual learning rate is multiplied by $\frac{b}{256}$, where b is the batch size.

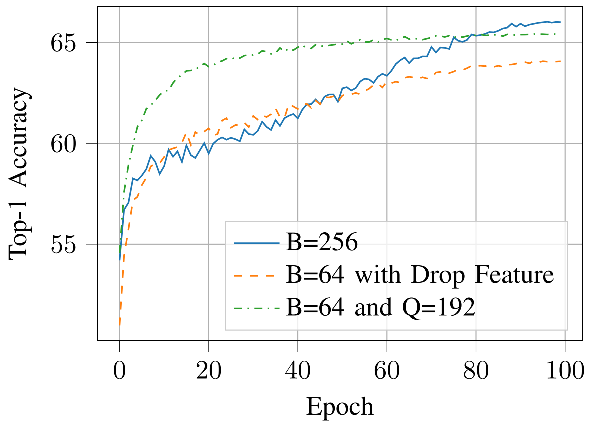

Fig. 1: Top-1 accuracy by epoch on the ImageNet validation set.

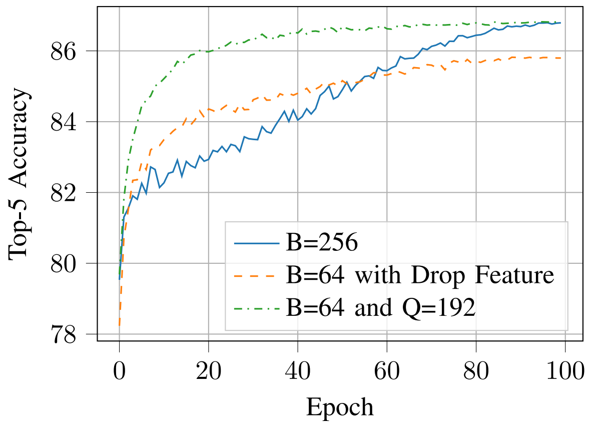

Fig. 2: Top-5 accuracy by epoch on the ImageNet validation set.

baseline.

The classification training loss is shown in Figure 4. Despite a similar performance in terms of classification accuracy, the baseline method ($B = 256$) presents a lower loss than both modifications ($B = 64$).

We also perform an experiment on the CIFAR-10 dataset [15]. In this case, the images are 32×32, and we remove solarization and Gaussian blur from the augmentations, but the augmentation settings remain the same otherwise. The input image channels are also normalized, using the mean and standard deviation values from ImageNet for simplicity.

For the CIFAR-10 experiments, we use a single machine with 1080 Ti 11 GB GPUs. The memory requirements are low in this case, so there is no economy in terms of hardware resources. We perform this experiment in order to verify that the use of the queue improves the accuracy of the model for small batch sizes.

Since the number of images in CIFAR-10 is much smaller than in ImageNet (60000 in CIFAR-10 and more than one million in ImageNet), we use a simpler architecture: the base network is ResNet-18, and the three fully connected layers in the projector have dimensions 512×2048, 2048×2048 and 2048×2048. To adapt the network to images with

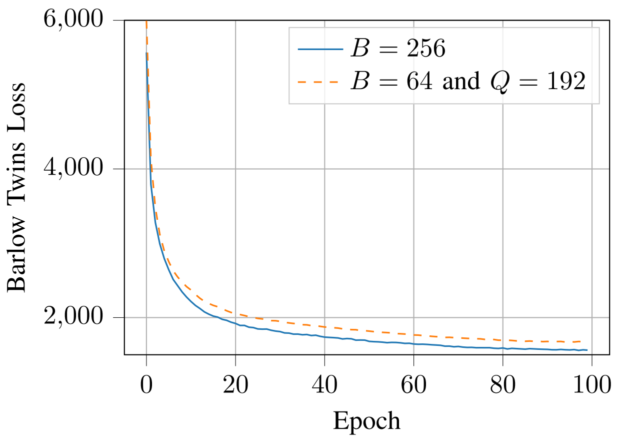

Fig. 3: Comparison of the Barlow Twins loss by epoch, on the ImageNet experiment, between the original method with batch size 256 and the queue method with a batch size of 64 and a queue of size 192. Since the number of dimensions of the output vector and the number of samples used to compute the cross-correlation matrix on the drop features method is different from the other methods, the loss is not directly comparable and is, thus, not exhibited here.

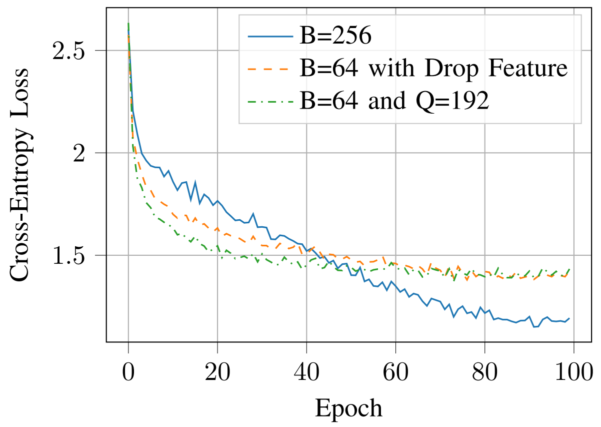

Fig. 4: Training cross-entropy loss of the fully connected classification layer.

smaller resolutions, the first layer, a 7×7 convolutional layer, is replaced by a 3×3 convolutional layer, and the 2×2 max-pooling layer that follows this first layer is removed. These modifications are the same as the ones in the MoCo-v2 implementation for CIFAR-10 [19].

We train four models with self-supervision on the CIFAR-10 train set for 800 epochs with SGD. Three with the baseline method and batch sizes of, 128, 32, and 16, with learning rates of, respectively, 0.001, 0.00025, and 0.000125, and one with a batch size of 16, a queue of size 112 (simulating a batch size of 128) and learning rate of 0.000125. The weight decay is set to 5×10^{-4}, the momentum is set to 0.9 and $\lambda = 0.0051$ for all models. The learning rate follows a cosine decay schedule, without warm-up.

For classification, we remove the projector network and replace it with a 512×10 fully connected layer, since we train to classify the 10 CIFAR-10 classes. The weights of the

TABLE II: Accuracy results on the CIFAR-10 test set after finetuning the classification layer. The batch size is B and the queue size is Q.

Method	Accuracy
Barlow Twins ($B = 16$)	87.0%
Barlow Twins ($B = 32$)	89.6%
Barlow Twins ($B = 128$)	90.0%
Barlow Twins ($B = 16$) with queue ($Q = 112$)	89.8%

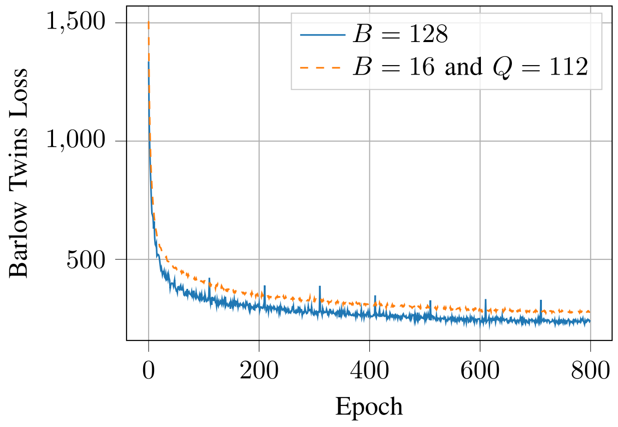

Fig. 5: Comparison of the Barlow Twins loss by epoch, on the CIFAR-10 experiment, between the original method with batch size 128 and the queue method with a batch size of 16 and a queue of size 112.

base network are frozen and we train the fully connected layer with the cross-entropy loss on the CIFAR-10 train set for 100 epochs with SGD. The learning rate is 0.3, the weight decay is 10^{-6}, the momentum is 0.9, and we use the cosine decay schedule.

The results of this method are summarized in Table II. In this experiment, the best performing method is the baseline with a batch size of 128, which has an accuracy of 90.0%. The inclusion of the queue improves the accuracy with batch size 16 from 87% to 89.8%, reaching a performance similar to the baseline method on larger batch sizes.

Similar to the ImageNet experiment, there is a gap, this time of around 14.3% between the loss with batch size of 16 and queue of 112 and the loss with batch size of 128. This can be seen in Figure 5.

V. Conclusions

Although the method that drops features does not reach the same performance as the original method on ImageNet, it does converge with batch size 64 with a reasonable performance. It also has the potential to be used in cases of higher dimensional outputs to decrease the memory requirements.

The queue method performs well, only slightly inferior to the original method on ImageNet. If this performance gap does not increase in longer trainings (with more epochs), then one might investigate whether the results of the original Barlow Twins paper can be approximated, with more modest hardware requirements.

In particular, these improvements allow us to train models with good performance on images with 224×224 resolution using only two 1080 Ti 11 GB GPUs, whereas more than four such GPUs are needed to train with the standard Barlow Twins method with a 256 batch size.

References

[1] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, “Barlow Twins: Self-supervised learning via redundancy reduction,” arXiv preprint arXiv:2103.03230, 2021.

[2] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, “A simple framework for contrastive learning of visual representations,” in Proceedings of the International Conference on Machine Learning (ICML), 2020.

[3] 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 (CVPR), June 2020.

[4] X. Chen and H. Fan, “Improved baselines with momentum contrastive learning,” arXiv preprint arXiv:2003.04297, 2020.

[5] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, “Unsupervised learning of visual features by contrasting cluster assignments,” in NeurIPS, 2020.

[6] J. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Pires, Z. Guo, M. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, “Bootstrap your own latent - A new approach to self-supervised learning,” in NeurIPS, 2020.

[7] A. Bardes, J. Ponce, and Y. LeCun, “VICReg: variance-invariance-covariance regularization for self-supervised learning,” in Proceedings of the International Conference on Learning Representations (ICLR), 2022.

[8] J. Deng, W. Dong, R. Socher, L. 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.

[9] 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.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in Proceedings of the International Conference on Computer Vision (ICCV), 2017.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” arXiv preprint arXiv:2010.11929, 2020.

[12] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked Autoencoders Are Scalable Vision Learners,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[13] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging Properties in Self-Supervised Vision Transformers,” in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

[14] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P. Huang, S. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, “DINOv2: Learning Robust Visual Features without Supervision,” arXiv preprint arXiv:2304.07193, 2023.

[15] A. Krizhevsky, “Learning Multiple Layers of Features from Tiny Images,” 2009, `https://www.cs.toronto.edu/~kriz/cifar.html`.

[16] H. Hotelling, “New light on the correlation coefficient and its transforms,” Journal of the Royal Statistical Society: Series B (Methodological), vol. 15(2), pp. 193-225, July 1953.

[17] S. Ioffe, and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, 2015.

[18] Y. You, I. Gitman, and B. Ginsburg, “Large batch training of convolutional networks,” arXiv preprint arXiv:1708.03888, 2017.

[19] `https://colab.research.google.com/github/facebookresearch/moco/blob/colab-notebook/colab/moco_cifar10_demo.ipynb`.