S^2-BNN: Bridging the Gap Between Self-Supervised Real and 1-bit Neural Networks via Guided Distribution Calibration

Zhiqiang Shen[1], Zechun Liu[1,2], Jie Qin[3], Lei Huang[3], Kwang-Ting Cheng[2], Marios Savvides[1]
[1]Carnegie Mellon University [2]Hong Kong University of Science and Technology
[3]Inception Institute of Artificial Intelligence
{zhiqians,zechunl,marioss}@andrew.cmu.edu
{qinjiebuaa,huanglei36060520}@gmail.com timcheng@ust.hk

Abstract

*Previous studies dominantly target at self-supervised learning on real-valued networks and have achieved many promising results. However, on the more challenging binary neural networks (BNNs), this task has not yet been fully explored in the community. In this paper, we focus on this more difficult scenario: learning networks where both weights and activations are binary, meanwhile, without any human annotated labels. We observe that the commonly used contrastive objective is not satisfying on BNNs for competitive accuracy, since the backbone network contains relatively limited capacity and representation ability. Hence instead of directly applying existing self-supervised methods, which cause a severe decline in performance, we present a novel guided learning paradigm from real-valued to distill binary networks on the final prediction distribution, to minimize the loss and obtain desirable accuracy. Our proposed method can boost the simple contrastive learning baseline by an absolute gain of **5.5∼15%** on BNNs. We further reveal that it is difficult for BNNs to recover the similar predictive distributions as real-valued models when training without labels. Thus, how to calibrate them is key to address the degradation in performance. Extensive experiments are conducted on the large-scale ImageNet and downstream datasets. Our method achieves substantial improvement over the simple contrastive learning baseline, and is even comparable to many mainstream supervised BNN methods. Code is available at https://github.com/szq0214/S2-BNN.*

1. Introduction

The recent advances and breakthroughs in 1-bit convolutional neural networks (1-bit CNNs), also known as binary neural networks [7, 30] mainly lie in supervised learning [21, 23]. With the binary nature of BNNs, such networks have been recognized as one of the most efficient and promising deep compression techniques for deploying models in resource-limited devices. Generally, as introduced in [30], BNNs can produce up to 32× compressed memory and 58× practical computational reduction on a CPU or mobile device. Considering the immense potential of being directly deployed in intelligent devices or low-power hardware, it is well worth further studying the behaviors of self-supervised BNNs (**S^2-BNN**), i.e., BNNs without human-annotated labels, both to better understand the properties of BNNs in academia, as well as to extend the scope of their usage in industry and real-world applications.

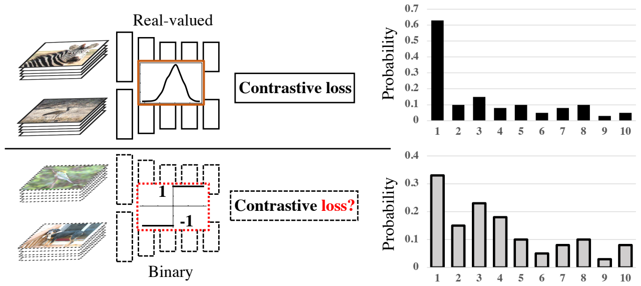


Figure 1. Illustration of the motivation for self-supervised BNNs. As the representation capability of BNNs is relatively limited, the output prediction is less confident than that of real-valued networks. Hence, we argue that the conventional self-supervised methods in real-valued networks may not be optimal for BNNs.

The goal of this paper is to study the mechanisms and properties of BNNs under the self-supervised learning scenario, then deliver practical guidelines on how to establish a strong self-supervised framework for them. To achieve this purpose, we start from exploring the widely used self-supervised contrastive learning in real-valued networks. Hence, our first question in this paper is: *Is the well-performing contrastive learning in real-valued networks still suitable for self-supervised BNNs?* Intuitively, binary networks are quite different from real-valued networks on both learning optimization and back-propagation of gradients since the weights and activations in BNNs are dis-

Table 1. A brief overview of our improvement over a variety of different architectures for binarizing models. We choose XNOR Net [30], Bi-Real Net [22] and ReActNet [21] as our backbone networks.

	XNOR (Re-impl.) [30]		Bi-Real Net [22]		ReActNet [21]	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
Supervised BNN	51.200	73.200	56.400	79.500	69.400	88.600
Contrastive Learning (MoCo V2) - Real-valued	–	–	50.296	75.206	60.776	82.830
Contrastive Learning (MoCo V2) - BNN (Baseline)	23.880	44.690	42.816	67.712	46.922	70.712
Contrastive Learning w/ Adam + lite aug. + progressive binarizing etc. (Ours) - BNN	–	–	–	–	52.452	76.080
+ Guided Learning (Ours) - BNN	–	–	–	–	56.022	79.168
Guided Learning Only (Ours) - BNN	**36.996**	**61.416**	**51.242**	**75.890**	**61.506**	**83.512**

crete, causing dissimilar predictions between the two different types of networks, as illustrated in Fig. 1. We answer this question by exploring the optimizers (SGD or the adaptive Adam optimizer), learning rate schedulers, data augmentation strategies, etc., and give optimal designs for self-supervised BNNs. These non-trivial studies enable us to build a base solution which brings about 5.5% improvement over the naïve contrastive learning of the baseline.

Subsequently, we empirically observe that the real-valued networks always achieve much better performance than BNNs on self-supervised learning (the comparison will be given later). Many recent studies [21, 23] have shown that BNNs demonstrate sufficient capability to achieve accuracy as high as the real-valued counterparts in supervised learning, but an appropriate learning strategy is required to unleash the potential of binary networks. Our second question is thus: *What are the essential causes for the performance gap between real-valued and binary neural networks in self-supervised learning?* It is natural to believe that if we can expose the causes behind the inferior results and also find a proper method for training self-supervised BNNs to mitigate the obliterated/poor accuracy, we can categorically obtain more competitive performance for self-supervised BNNs. Our discovery on this perspective is interesting: we observe that the distributions of predictions from BNNs and real-valued networks are significantly different but after using a frustratingly simple method through a *teacher-student* paradigm to calibrate the latent representation on BNNs, the performance of BNNs can be boosted substantially, with an extra $\sim$4% improvement.

Concretely, to address the issue of how to maximize the representation ability of self-supervised BNNs, we propose to add an additional self-supervised real-valued network branch to guide the target binary network learning. This is somewhat like knowledge distillation but the slight difference is that our teacher is a self-supervision learned network and the class for the final output is agnostic. We force the BNNs to mimic the final predictions of real-valued models after the projection MLP head and the softmax operation. In our framework, we introduce a strategy that enables the BNNs to mimic the distribution of a real-valued reference network smoothly. This procedure is called guided distillation in our method. Combining contrastive and guided learning is a spontaneous idea for tackling this problem, while intriguingly, we further observe that solely employing guided learning without contrastive loss can dramatically boost the performance of the target model by an additional 5.5%. This is surprising since, intuitively, combining both of them seems a better choice. To shed further light on this observation, i.e., contrastive learning is not necessary for directly training self-supervised BNNs, we study the learning mechanism behind contrastive and guided/distilled techniques and derive the insights that contrastive and guided learning basically focus on different aspects of feature representations. Distillation forces BNNs to mimic the reference network's predictive probability, while contrastive learning tries to discover and learn the latent patterns from the data itself. This paper does not argue that learning the isolated patterns by contrastive learning is not good, but from our experiments, it shows that recovering knowledge from a well-learned real-valued network with extremely high accuracy is more effective and practical for self-supervised BNNs. An overview of our improvement over various architectures is shown in Table 1.

To summarize, our contributions in this paper are:

- We are the first to study the problem of self-supervised binary neural networks. We provide many practical designs, including optimizer choice, learning rate scheduler, data augmentation, etc., which are useful to establish a base framework of self-supervised BNNs.
- We further propose a guided learning paradigm to boost the performance of self-supervised BNNs. We discuss the roles of contrastive and guided learning in our framework and study the way to use them.
- Our proposed framework improves the naïve contrastive learning by **5.5$\sim$15%** on ImageNet, and we further verify the effectiveness of our learned models on the downstream datasets through transfer learning.

2. Related Work

Binary Neural Networks. Binary neural networks [7, 30, 20, 22, 29, 23, 21] have been widely studied in the recent years. The first work can be traced back to EBP [34] and BNNs [7]. After that, many interesting explorations have emerged. XNOR Net [30] is a representative study that proposed the real-valued scaling factors for multiplying with each of binary weight kernels, this method has become a commonly used binarization strategy in the community

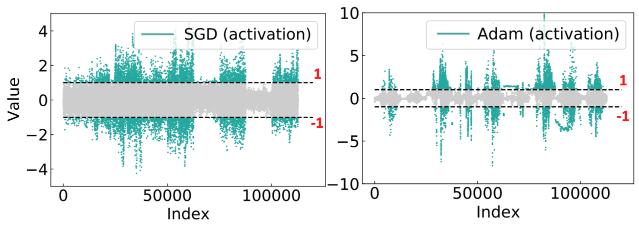


Figure 2. Illustration of activation distributions from different optimizers (Left: SGD; Right: Adam). Dotted lines are the up (+1) and low (-1) bounds. We observe that Adam can alleviate activation saturation significantly during training.

and boosted the accuracy of BNNs significantly. Real-to-binary [23] adopted the better training scheme and attention mechanism to propagate binary operation on the activations and obtained better accuracy. ReActNet [21] further studied the non-linear activations for BNNs and built a strong baseline upon MobileNet [18]. The proposed method achieved fairly competitive performance on large-scale ImageNet.

Self-supervised Learning. Self-supervised learning (SSL) is a technique that aims to learn the internal distributions and representations automatically through data, meanwhile, without involving any human annotated labels. Early works mainly stemmed from reconstructing input images from a latent representation, such as auto-encoders [36], sparse coding [26], etc. Following that, more and more studies focused on exploring and designing handcrafted pretext tasks, such as image colorization [40], jigsaw puzzles [25], rotation prediction [12], pretext-invariant representations [24], etc. Recently, contrastive based visual representation learning [15] has attracted much attention in the community and achieved breakthroughs and promising results. Among them, MoCo [16] and SimCLR [5] are two representative methods emerged recently. Also, many interesting works [27, 17, 1, 35, 32, 13, 4] have been proposed. In this paper, we expose that the distillation process from a self-supervised strong teacher to the efficient binary student is more effective than learning binary student directly using contrastive learning. A concurrent study SEED [11] also employed self-supervised distillation loss, which can be considered as a contemporaneous work of ours.

Self-supervised Learning on BNNs. To the best of our knowledge, there are no existing works focusing on exploring BNNs with self-supervised scheme. The proposed approach in this paper has quite appealing advantages on this direction. We will elaborate and validate the proposed method in the following sections. In the network quantization area, Vogel et al. [37] presented a non-retraining method for quantizing networks. This may be the closest work to our study. However, they used the intermediate features of the network based on the valid input samples to supervise the quantization procedure, which is not related to this work, also entirely different from the perspective of our contrastive based or guided learning paradigms.

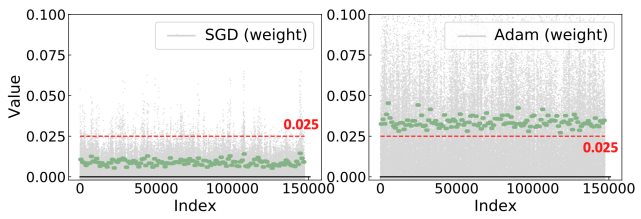


Figure 3. Visualization of the weight distributions in the first layer. For clarity, we use *green* hyphens to mark the l_1 norm of weights in each channel. The *red* dotted line denotes the reference value (0.025) of weights.

3. Optimizer Effects of SSL on BNNs

Saturation on Activations and Gradients. We first introduce an activation saturation phenomenon in the self-supervised BNNs scenario. When the absolute values of activations exceed one, the corresponding gradients are suppressed to be zero by the formulation of approximation in the derivative of the *sign* function [9]. We study this observation for explaining why the optimizer used in self-supervised methods, e.g., MoCo [16] with SGD performs well for real-valued networks, but it is not optimal on binary networks. This exploration can help us determine which optimizer is superior for our proposed self-supervised method. Upon our observation, activation saturation emerges in most layers of a binary network and it always affects the magnitude of gradients critically on different channels. As shown in Fig. 2, we visualize the activation distributions of the first binary convolution layer of our networks. We can observe that, for the particular input batch images, a large number of activations exceed the bounds of -1 and +1, which causes the gradient passing those neurons to be zero-valued and makes more weights less active.

Different Optimizer Effects. Adam adapts the learning rate according to the historical values and amplifies the gradients with small ones. The strength of Adam stems from the regularization effect of second-order momentum, which is crucial to revitalize the inactive weights, i.e., zero-valued ones due to the activation saturation in BNNs as introduced above. Interestingly, Adam can rouse most of the weights to be active again with better generalization ability. The visualization of weight distribution between SGD and Adam in the first layer is shown in Fig. 3. The red dotted lines are references at the value of 0.025. The green poly-lines are the l_1 norm values of weights in each output channel for better comparing to the numerical value between SGD and Adam. It is obvious that Adam contains overwhelmingly larger weights than SGD, which reflects the weights optimized by SGD are not as good as those with Adam.

In contrast to the SGD optimizer that only accumulates the first momentum, the adaptive method Adam, predominantly uses the accumulation in the second momentum to amplify the learning rate regarding the gradients with small historical values. SGD with momentum updating is used to

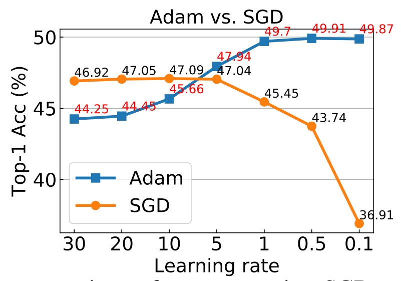


Figure 4. A comparison of accuracy using SGD and Adam for training backbone networks. The learning rates shown in the figure are in linear evaluation stage and we explore the optimal configurations for them to show the best capability of the two optimizers.

help accelerate and dampen oscillations on gradients, it can be formulated as: $m_t = \beta m_{t-1} + \eta \nabla_\theta J(\theta)$, $\theta = \theta - m_t$, where $\nabla_\theta J(\theta)$ is the gradient and β is exponential rate. The updating rule in Adam is defined as: $\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t$, $\hat{m}_t$ and $\hat{v}_t$ denote exponential moving averages of the gradient and the squared gradient, respectively. With $\hat{m}_t$ drawing $\hat{v}_t$ of the uncentered gradient variance, the update value is normalized to alleviate the discrepancy in the gradient. Fig. 4 shows the accuracy comparisons on the linear evaluation stage with SGD and Adam-trained backbones in self-supervised learning. It can be observed that, with SGD training, the accuracy decreases when learning rates become smaller. This tendency is consistent with the real-valued model. While with Adam, the accuracy increases dramatically when using smaller learning rates, and the best final accuracy is much higher than the best result from SGD.

4. Data Augmentation Adjustments

Our data augmentation strategies mainly inherit from the baseline method MoCo V2 [6]. In real-valued networks, heavier augmentations have been proven useful in most cases of contrastive based self-supervised learning. However, considering the limited capability of BNNs to distinguish the same class from different shapes of images, instead of involving more data augmentations, we decrease the transformations' probabilities of *ColorJitter* and *GaussianBlur* to facilitate the difficulty for BNNs to classify the two images in the same class. Intriguingly, this lite data augmentation strategy can bring an additional ∼1.0% improvement on ImageNet. This reflects that the properties of BNNs are basically different from the real-valued networks, thus the configurations are required to be reconsidered. It also demonstrates the value of this study on exploring self-supervised BNNs. More details are provided in Sec. 6.1.

5. Our Approach

Our roadmap of this paper has three main stages: Firstly, we follow the real-valued self-supervised method with contrastive loss whereas replacing particular configurations to fit the properties of BNNs, such as optimizer, data augmentation, learning rate, etc. These strategies can produce 5.5% improvement over vanilla MoCo V2 baseline. Then, we propose to adopt an additional guided learning method to enforce representations of BNNs to be similar to the real-valued reference network. This simple strategy can bring an additional ∼4% improvement. Lastly, we remove contrastive loss and solely optimize BNNs with the guided learning paradigm and the performance is further increased by 5.5%. Several motivations and insights of our proposed method are discussed in the following sections.

5.1. Preliminaries

BNNs aim to learn networks that both weights and activations are with discrete values in {-1, +1}. In the forward propagation of training, the real-valued activations will be binarized by the sign function: $\mathcal{A}_b = \text{Sign}(\mathcal{A}_r) = \begin{cases} -1 & \text{if } \mathcal{A}_r < 0, \\ +1 & \text{otherwise.} \end{cases}$ where $\mathcal{A}_r$ is the real-valued activation of the previous layers calculated from the binary or real-valued convolutional operations. $\mathcal{A}_b$ is the binarized activation. The real-valued weights in the model will be binarized through: $\mathbf{W}_b = \frac{||\mathbf{W}_r||_{l_1}}{n}\text{Sign}(\mathbf{W}_r) = \begin{cases} -\frac{||\mathbf{W}_r||_{l_1}}{n} & \text{if } \mathbf{W}_r < 0, \\ +\frac{||\mathbf{W}_r||_{l_1}}{n} & \text{otherwise.} \end{cases}$ where $\mathbf{W}_r$ is the real-valued weights that are maintained as *latent* parameters to accumulate the tiny gradients, n is #weight in each channel. $\mathbf{W}_b$ is the weights after binarization. The binary weights will be updated through multiplying the sign of latent real-valued weights and the channel-wise l_1 norm $(\frac{1}{n}||\mathbf{W}_r||_{l_1})$. The gradient g is calculated with respect to binary weights $\mathbf{W}_b$: $g_t = \nabla_{\mathbf{W}_{b_t}} J(\mathbf{W}_{b_t}) \cdot 1_{|\mathbf{W}_{r_t}|<1}$, where t is #iteration.

Training BNNs is a challenging task since the gradient for optimizing parameters in the network is approximated and the capacity of models for memorizing all data distributions is also limited. It is thus worthwhile to discuss that as the *sign* function has a bounded range, the approximation to the derivative of the *sign* operation will suffer from a vanished gradient issue when the activations exceed the effective gradient range, i.e., $[-1, 1]$.

5.2. Real-Value Guided Distillation

Self-supervised Contrastive Loss. The conventional contrastive learning uses a standard log-softmax function to apply one positive sample out of K negative samples and it predicts the probability of data distribution as:

$$\mathcal{L}_{CL} = -\log \frac{\exp\left(s\left(\mathbf{v}_I, \mathbf{v}_{\hat{I}}\right)/\tau\right)}{\exp\left(s\left(\mathbf{v}_I, \mathbf{v}_{\hat{I}}\right)/\tau\right) + \sum_{\hat{I}' \in \mathbf{Neg}} \exp\left(s\left(\mathbf{v}_I, \mathbf{v}_{\hat{I}'}\right)/\tau\right)} \tag{1}$$

where $\left(\mathbf{v}_I, \mathbf{v}_{\hat{I}}\right)$ are two random "views" of the same image under random data augmentation. s is the cosine similarity or other matching function for measuring the similarity of

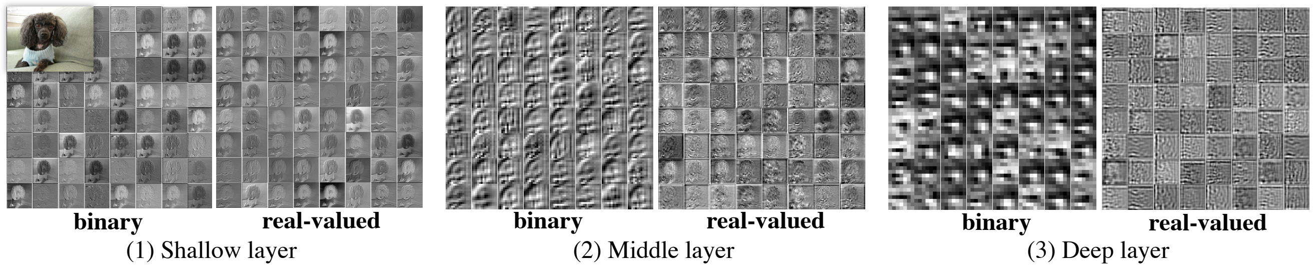


Figure 5. Illustration of activation distributions on shallow, middle and deep layers of self-supervised binary and real-valued networks. The input image is in the upper left corner of the first subfigure. Interestingly, we observe that at the shallow layer, the semantic representations between these two models are visually similar, while on the middle and deep layers, real-valued representations obviously contain richer information than BNNs in the activation maps, which is beneficial for learning good latent features. This phenomenon motivates us to propose a method for calibrating the high-level distribution of BNNs in a self-supervised learning scenario to arouse its potential.

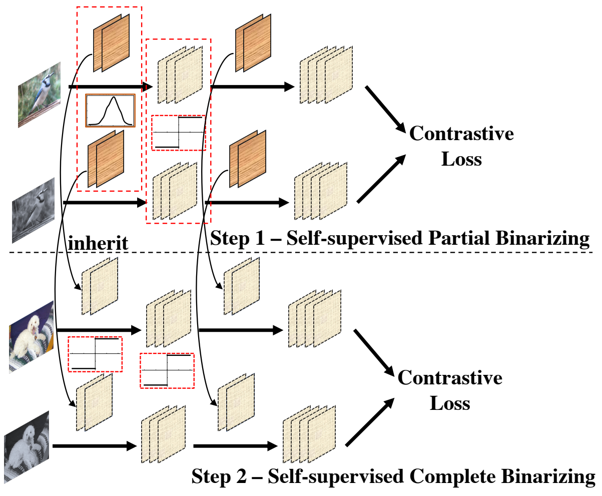

Contrastive Loss
inherit
Step 1 – Self-supervised Partial Binarizing
Contrastive Loss
Step 2 – Self-supervised Complete Binarizing

Figure 6. Illustration of progressive binarization. We take the contrastive based framework as an example, while it is also feasible for the proposed guided learning solely framework.

two representations. τ is a temperature hyper-parameter.

Guided with KL-divergence Loss. KL-divergence loss is used to measure the degree of how one probability distribution is different from another reference one. We train the BNNs Bi_θ by minimizing the KL-divergence between its output $\mathbf{p}^{\mathrm{Bi}_\theta}(x_i)$ and the representation $\mathbf{p}^{\mathrm{Real}_\theta}(x_i)$ generated by a self-supervised real-valued reference model. The loss function can be formulated as:

$$\mathcal{L}_{KL}(\mathrm{Bi}_\theta) = -\frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}^{\mathrm{Real}_\theta}(x_i)/\tau)\log(\frac{\mathbf{p}^{\mathrm{Bi}_\theta}(x_i)/\tau}{\mathbf{p}^{\mathrm{Real}_\theta}(x_i)/\tau}) \tag{2}$$

where N is the number of samples. τ is a temperature hyper-parameter. Note that the data augmentation strategy should be the same for both binary and real-valued models. In practice, we only optimize with cross-entropy loss as:

$$\mathcal{L}_{CE}(\mathrm{Bi}_\theta) = -\frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}^{\mathrm{Real}_\theta}(x_i)/\tau)\log(\mathbf{p}^{\mathrm{Bi}_\theta}(x_i)/\tau) \tag{3}$$

which is equivalent to $\mathcal{L}_{KL}$ following MEAL [31, 33].

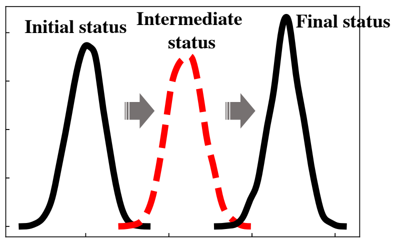
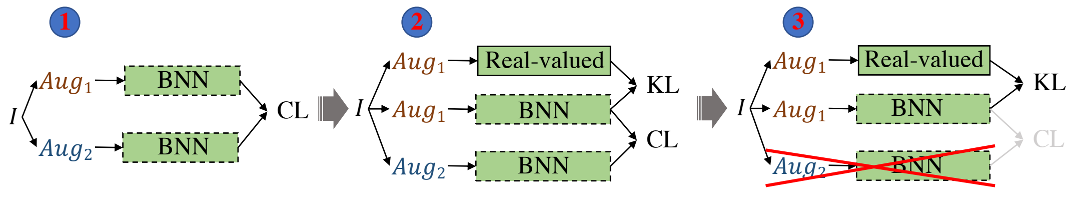


Figure 7. Illustration of binarization with intermediate status to facilitate self-supervised learning difficulties.

5.3. Progressive Binarization

As illustrated in Fig. 5, there are many differences on activation distributions between binary and real-valued networks across middle and high-level layers, and real-valued activations always contain more fine-grained details and semantic information of instance and background on representation distributions. As our purpose is to recover the distributions from real-valued networks to binary networks, we propose to adopt a multi-step binarization procedure. The motivation behind this design is straight-forward, as shown in Fig. 7, directly recovering distribution from real-valued networks to binary networks is challenging, to facilitate the difficulty of optimization, we first keep partial parameters or weights in the target model to be real-valued, and then binarize them progressively. This strategy is somewhat similar to [23, 21], while we emphasize that all these previous studies lie in supervised learning, here our objective is a self-supervised contrastive loss or distillation loss. Hence the learning procedure and hyper-parameter design are entirely different from prior works.

In our method, the initial status is a complete real-valued network, the intermediate status is a partially binarized network with real-valued weights and binary activations, as shown in Fig. 6. We train such a network first to obtain the real-valued parameters, then we reuse these pre-trained parameters in the final completely binary models. Since the binarization is modeled by the *sign* function during training, hence the binary models can inherit the real-valued parameters as the initialization. This study shows that such a



(1) Basic (2) CL + KL (proposed) (3) KL (proposed)

Figure 8. Illustration of three strategies of our frameworks. "CL" denotes the conventional contrastive learning and "KL" denotes Kullback–Leibler divergence, i.e. the proposed guided learning. In each subfigure, solid and dashed boxes represent freezing and non-freezing parameters in training, respectively.

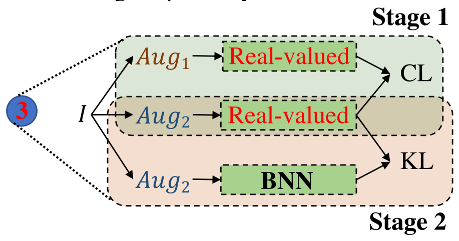


Figure 9. Illustration of online and offline training of self-supervised real-valued reference network. This is also a more detailed explanation of ③ in Fig. 8.

multi-step binarization can facilitate optimization for self-supervised training and obtain significant improvement.

Weight Decay Strategy. Weight decay is a widely used technique for preventing networks from overfitting. We observe that it is necessary to adopt an appropriate weight decay strategy in different steps of our multi-step training. Since the intermediate status is only a transitional phase, the existing real-valued weights can make model's capacity larger than the final completely-binary status, thus the weight decay is not employed in the first step (or choosing a smaller value of weight decay in this phase). For the second step, weight decay is adopted to avoid overfitting.

5.4. What Happens If Removing Contrastive Loss

Since adding guided learning term brings a substantial improvement, we are curious whether guided learning is capable of learning good representations solely. Our observation is surprising, removing contrastive loss gives an additional 5.5% improvement. We conjecture this is because contrastive and guided learning are basically optimizing with different directions. Guided learning is mimicking the real-valued high-quality representation and recovering the knowledge stored in it, if the reference model is strong enough, the target BNNs can be extremely well-performing, while contrastive loss learns from data itself which explores different patterns (e.g., instance discrimination, colorization, etc.) from guided learning. Therefore, in this work we study the following **three schemes**, as shown in Fig. 8:

①: Enhanced baseline of contrastive learning.

②: Contrastive + guided learning (distillation).

③: Guided learning (distillation) only.

Where is the self-supervised real-valued network from? There are two ways to obtain the real-valued reference network: (1) online training with the target BNN simultaneously; (2) offline pretraining. As shown in Fig. 9, if we train "stage 1" and "stage 2" together, this is the online training scheme and the real-valued network will be optimized together with binary network. However, the learning cost will increase significantly since in each individual run we have to include an additional real-valued branch. A simpler and more efficient way is to train "stage 1" offline and in advance, then reuse it for all experiments. The offline strategy is utilized in this work of all our experiments. Based on the perspective of MEAL V2 [33] that better teachers usually distill better students, we choose MoCo V2 800-ep pretrained real-valued ResNet-50 as our strong teacher model.

Algorithm 1 Online and offline training for self-supervised real-valued reference and target binary networks.

```
Preparation:
  1: $x_1 = aug_1(x)$                                  ▷ $x$ is the input image
  2: $x_2 = aug_2(x)$
  3: $\mathbf{p}_1^{\text{Real}_\theta} = f_{\text{Real}_\theta}(x_1)$        ▷ $f_{\text{Real}_\theta}$ is the real-valued network
  4: $\mathbf{p}_2^{\text{Real}_\theta} = f_{\text{Real}_\theta}(x_2)$
  5: $\mathbf{p}^{\text{Bi}_\theta} = f_{\text{Bi}_\theta}(x_2)$              ▷ $f_{\text{Bi}_\theta}$ is the binary network
Online Scheme:
  1: $\mathcal{L}(f_{\text{Bi}_\theta}, f_{\text{Real}_\theta}) = \mathcal{L}_{CL}(\mathbf{p}_1^{\text{Real}_\theta}, \mathbf{p}_2^{\text{Real}_\theta}) + \mathcal{L}_{KL}(\mathbf{p}^{\text{Bi}_\theta}, \mathbf{p}_2^{\text{Real}_\theta})$
Offline Scheme:
  1: Stage 1: $\mathcal{L}(f_{\text{Real}_\theta}) = \mathcal{L}_{CL}(\mathbf{p}_1^{\text{Real}_\theta}, \mathbf{p}_2^{\text{Real}_\theta})$
  2: $\mathbf{p}_2^{\text{Real}_\theta} = f_{\text{Real}_\theta}(x_2)$
  3: $\mathbf{p}^{\text{Bi}_\theta} = f_{\text{Bi}_\theta}(x_2)$
  4: $f_{\text{Real}_\theta} = f_{\text{Real}_\theta}$.detach()                   ▷ stop-gradient()
  5: Stage 2: $\mathcal{L}(f_{\text{Bi}_\theta}) = \mathcal{L}_{KL}(\mathbf{p}^{\text{Bi}_\theta}, \mathbf{p}_2^{\text{Real}_\theta})$
```

6. Experiments

In this section, we first introduce the datasets we used and implementation details for self-supervised pre-training, linear evaluation and transfer learning. Then, we provide extensive ablation studies for each component in our method. Following that, we show our main and transfer results. Lastly, we illustrate activation visualizations to further demonstrate the effectiveness of our proposed method.

6.1. Datasets and Implementation Details

Datasets. Our experiments are conducted on the widely-used large-scale ImageNet 2012 dataset [8], which contains 1,000 classes with a total number of 1.2 million training images and 50,000 images for validation. For transfer learning, we use PASCAL VOC2007 [10], CUB200-2011 [38], Birdsnap [2] and CIFAR-10/100 [19] benchmarks.

Data Augmentation. As mentioned above, our basic data augmentation follows MoCo V2 [6] with no additional operations, but we reduce the probability of *ColorJitter* from 0.8 to 0.6 and *GaussianBlur* from 0.5 to 0.2. We apply this lite data augmentation strategy to all of our experiments.

Self-supervised Pre-training. We adopt MoCo V2 [6] as the baseline self-supervised method. For our distillation solution, we use *none* of momentum update, shuffling BN, a memory bank (negative pairs) and contrastive loss. The initial learning rates are 3×10^{-2} for SGD following [6] and 3×10^{-4} for Adam, and are reduced with a linear decay through *lr* = (initial *lr*) × (1 - epoch / total_epoch). τ is set to 0.2 for both contrastive and distillation losses. If no otherwise specified, all networks are trained with 200 epochs.

Linear Evaluation. We freeze all the parameters in the backbone and train a supervised linear classifier using the conventional self-supervised evaluation protocol [6, 5, 32]. We train with smaller *lr* and 100 epochs, and other hyper-parameters are following the baseline method [6].

Transfer Learning. We fine-tune the entire network using the weights of our learned models as initializations. We train for 180 epochs with a batch size of 128 and an initial learning rate of 0.01. On PASCAL VOC multi-object classification task, we adopt sigmoid cross-entropy instead of softmax one. We use SGD with a momentum parameter of 0.9 and weight decay of 0.0001. We perform standard random crops with resize and flips as data augmentation during fine-tuning. The training image size is 224×224. At test time, we resize images to 256 pixels and take a 224×224 center crop. When freezing backbone, we solely train the last linear layer as the standard linear evaluation protocol.

6.2. Ablation Studies

Optimizers. We study the standard SGD and adaptive optimizer Adam in the pre-training stage. Our results in Fig. 4 shows that Adam can bring about 2.8% improvement.

Learning Rate Scheduler. Here the learning rate scheduler indicates in the linear evaluation stage. In the training stage, we use a uniform value of 3×10^{-4} as presented above. The results are shown in Table 2, we provide results of our three schemes on the *lr* range from 30 to 0.05. It can be observed 0.1 is the optimal choice for the Adam optimized models.

Data Augmentation Effects. We conducted a comparison using our proposed lite data augmentation and MoCo V2 vanilla strategy, and the same contrastive loss and Adam optimizer are used here. The Top-1/5 results are neatly improved from 49.402/73.152 to 50.410/73.968 on ImageNet.

Multi-step Binarization. Our results are shown in Table 3, the proposed multi-step strategy generally obtains better accuracy. Whereas, the improvement seems to decrease when the base performance becomes higher, i.e., from ① to ③.

Table 2. One-step results with different *lr* in linear evaluation.

lr	①	②	③
30	44.248	47.918	54.518
20	44.454	48.570	54.986
10	45.662	50.054	56.050
5	47.942	51.808	57.868
1	49.702	53.324	59.838
0.5	**49.914**	53.468	59.968
0.1	49.870	**53.484**	**60.418**
0.05	49.228	52.926	60.304

Table 3. Comparison of one-step and multi-step binarization.

	①	②	③
Complete in one step	49.914	53.484	**60.418**
Multi-step binarization	52.452	56.022	**61.506**

Table 4. Comparison of the Top-1 accuracy on ImageNet with supervised and self-supervised state-of-the-art methods.

Binary Methods		#Epoch	BOPs ($\times10^9$)	FLOPs ($\times10^8$)	OPs ($\times10^8$)	Acc (%) Top-1
Supervised Learning:						
BNNs [7]		–	1.70	1.20	1.47	42.2
XNOR-Net [30]		–	1.70	1.41	1.67	51.2
MobiNet [28]		–	–	–	0.52	54.4
Bi-RealNet-18 [22]		–	1.68	1.39	1.63	56.4
PCNN [14]		–	–	–	1.63	57.3
CI-BCNN [39]		–	–	–	1.63	59.9
Binary MobileNet [29]		–	–	–	1.54	60.9
Real-to-Binary [23]		–	1.68	1.56	1.83	65.4
MeliusNet29 [3]		–	5.47	1.29	2.14	65.8
ReActNet [21]		–	4.82	0.12	0.87	69.4
Self-Supervised Learning:						
MoCo V2 [6] (baseline)		200	4.82	0.12	0.87	46.9
	①	200	4.82	0.12	0.87	52.5
Ours	②	200	4.82	0.12	0.87	56.0
	③	200	4.82	0.12	0.87	**61.5**

Different Architectures and Strategies. The results with different backbones are shown in Table 1, we choose XNOR Net [30], Bi-Real Net [22] and ReActNet [21] as our backbones for this ablation study. It can be seen that we obtain substantial improvement across all of these architectures.

6.3. Main Results

A summary of our main results is shown in Table 4, we adopt ReActNet as our backbone network. Comparing to the self-supervised baseline MoCo V2, our method outperforms it by 14.6% with the same training epochs. Promisingly, it can be observed that our results are even comparable to some recently proposed supervised methods, such as Bi-RealNet-18 [22], CI-BCNN [39] while only containing about 1/2 OPs to them. The results demonstrate the great potential of our self-supervised BNN method on real-world applications where annotation and memory are both scarce.

Visualization. To better understand where the boost comes from in our distillation method, we further visualize the activation maps of contrastive and guided learned models at the same level of layers. As shown in Fig. 10, in each group,

Table 5. Transfer accuracy on the classification task.

	VOC2007	CUB200-2011	Birdsnap	CIFAR-10	CIFAR-100
From Scratch (Real-valued)	72.7	29.8	46.2	93.1	70.9
From Scratch (Binary)	50.0	–	–	65.9	37.2
Fine-tune:					
MoCo V2 Real-valued (baseline 1)	89.6	67.3	63.6	95.3	79.3
MoCo V2 Binary (baseline 2)	81.0	34.4	34.0	89.9	69.5
Ours (①)	82.3	38.2	38.0	91.5	71.9
Ours (②)	83.5	40.5	39.2	91.3	72.3
Ours (③)	**86.9**	**50.1**	**45.7**	**92.7**	**74.3**
Freeze backbone:					
MoCo V2 Real-valued (baseline 1)	86.5	51.5	22.8	86.9	60.7
MoCo V2 Binary (baseline 2)	79.8	23.3	20.3	79.3	56.7
Ours (①)	81.7	33.1	21.9	80.4	58.7
Ours (②)	83.1	38.4	25.6	80.7	58.8
Ours (③)	**86.4**	**47.5**	**34.1**	**82.7**	**61.9**

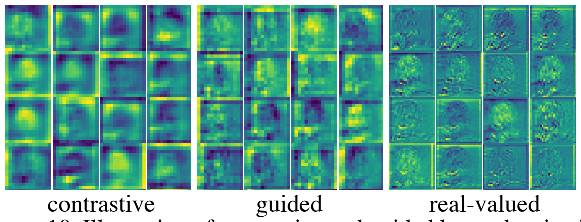


Figure 10. Illustration of contrastive and guided learned activation maps from the same layer of models. Visually, it can be identified that the quality of activation maps from contrastive learning to guided learning is improved with more details.

we visualize the first 16 channels in those layers. Visually, it can be recognized that the quality of activation maps from contrastive learning to guided learning is improved significantly with more details, and guided learning results are more close to the real-valued ones.

More Training Epochs. In real-valued scenario of self-supervised learning, more training budget always obtains a significant improvement. For example, SwAV [4] achieves 0.7% gain when training from 200 to 400 epochs. However, we also train our model with 400 epochs but we found the improvement is marginal (from 61.5% to 61.8%). We conjecture the reason is that our distillation based framework utilizes neither positive nor negative pairs, the binary student basically recovers the teacher's capability, so it is bounded by teacher's ability rather than the training budget.

Training Cost Analysis. Compared to the self-supervised baseline method, our main extra training cost is the learning procedure of generating the self-supervised real-valued model. As we adopt the offline strategy in our framework, we only need to train it once, hence if not considering this pre-training process, our total computational cost is nearly the same as the baseline MoCo V2.

Why Solely Using Distillation Loss is Better Than Combining with Contrastive Loss for Self-supervised BNNs. Intuitively, distillation loss forces BNNs to *mimic the reference network's predictive probability*, while contrastive learning tends to *discover the latent patterns from the data itself*. Our Fig. 10 evidences that in binary scenario, contrastive loss is relatively weaker than distillation loss to learn fine-grained representations, and the semantics are also vague. Combining both of them may not be an optimal solution due to the discrepancy of the optimization spaces.

Transfer Learning. It is critical to further verify the transferability of our learned parameters from different learning schemes. We follow the conventional self-supervised fine-tuning evaluation protocol for this study. The summary of our transfer results is provided in Table 5, all the network structures in this table are MobileNet-like ReActNet. Our results of ① can be regarded as the stronger contrastive baseline. "From scratch" denotes we train networks with the randomly initialized parameters and we show them here for the reference purpose. Generally, our transfer results are consistent with their linear evaluation performance on ImageNet. In particular, the improvement from ② to ③ is dramatically higher than that from ① to ② across different datasets. Moreover, we observe our best result is even close to the self-supervised real-valued baseline.

7. Conclusion

It is worthwhile considering how to train a robust and accurate self-supervised binary network. In this work, we have summarized and explained several behaviors observed while training such networks without labels. We focused on how optimizer, learning rate scheduler and data augmentation encourage representations and affect the performance in building a base BNN framework. We further proposed a guided learning paradigm enabled by a real-valued reference network to distill the target binary network training, and exposed such learning strategy can obtain better results comparing to both the contrastive learning and even supervised learning BNNs scheme. We attribute the proposed superior training scheme to its ability of mimicking the high quality of the reference network's representation. Finally, we performed extensive ablation experiments on each component of our method. Moreover, our trained parameters can be crucial for many downstream tasks that depend on a good representation, such as fine-grained recognition, etc.

References

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15535–15545, 2019. 3

[2] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2011–2018, 2014. 7

[3] Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, and Christoph Meinel. Meliusnet: Can binary neural networks achieve mobilenet-level accuracy? *arXiv preprint arXiv:2001.05936*, 2020. 7

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 3, 8

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3, 7

[6] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 4, 7

[7] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016. 1, 2, 7

[8] 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. 6

[9] Ruizhou Ding, Ting-Wu Chin, Zeye Liu, and Diana Marculescu. Regularizing activation distribution for training binarized deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11408–11417, 2019. 3

[10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 7

[11] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. *arXiv preprint arXiv:2101.04731*, 2021. 3

[12] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 3

[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, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 3

[14] Jiaxin Gu, Ce Li, Baochang Zhang, Jungong Han, Xianbin Cao, Jianzhuang Liu, and David Doermann. Projection convolutional neural networks for 1-bit cnns via discrete back propagation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8344–8351, 2019. 7

[15] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 3

[16] 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*, pages 9729–9738, 2020. 3

[17] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018. 3

[18] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3

[19] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009. 7

[20] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In *Advances in Neural Information Processing Systems*, pages 345–353, 2017. 2

[21] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *ECCV*, 2020. 1, 2, 3, 5, 7

[22] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, pages 722–737, 2018. 2, 7

[23] Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. In *ICLR*, 2020. 1, 2, 3, 5, 7

[24] Ishan Misra and Laurens van der 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. 3

[25] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 3

[26] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. 3

[27] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 3

[28] Hai Phan, Yihui He, Marios Savvides, Zhiqiang Shen, et al. Mobinet: A mobile binary network for image classification. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 3453–3462, 2020. 7

[29] Hai Phan, Zechun Liu, Dang Huynh, Marios Savvides, Kwang-Ting Cheng, and Zhiqiang Shen. Binarizing mobilenet via evolution-based searching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13420–13429, 2020. 2, 7

[30] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016. 1, 2, 7

[31] Zhiqiang Shen, Zhankui He, and Xiangyang Xue. Meal: Multi-model ensemble via adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4886–4893, 2019. 5

[32] Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, Trevor Darrell, and Eric Xing. Un-mix: Rethinking image mixtures for unsupervised visual representation learning. *arXiv preprint arXiv:2003.05438*, 3(7), 2020. 3, 7

[33] Zhiqiang Shen and Marios Savvides. Meal v2: Boosting vanilla resnet-50 to 80%+ top-1 accuracy on imagenet without tricks. *arXiv preprint arXiv:2009.08453*, 2020. 5, 6

[34] Daniel Soudry, Itay Hubara, and Ron Meir. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *Advances in neural information processing systems*, pages 963–971, 2014. 2

[35] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 3

[36] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008. 3

[37] S. Vogel, J. Springer, A. Guntoro, and G. Ascheid. Self-supervised quantization of pre-trained neural networks for multiplierless acceleration. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1094–1099, 2019. 3

[38] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 7

[39] Ziwei Wang, Jiwen Lu, Chenxin Tao, Jie Zhou, and Qi Tian. Learning channel-wise interactions for binary convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 568–577, 2019. 7

[40] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 3