ProxyBNN: Learning Binarized Neural Networks via Proxy Matrices

Xiangyu He[1,2], Zitao Mo[1], Ke Cheng[1,2], Weixiang Xu[1,2], Qinghao Hu[1], Peisong Wang[1], Qingshan Liu[4], and Jian Cheng[1,2,3]✉[0000−0003−1289−2758]

[1] NLPR, Institute of Automation, Chinese Academy of Sciences
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences
[3] Center for Excellence in Brain Science and Intelligence Technology, Beijing, China
{xiangyu.he, qinghao.hu, peisong.wang, jcheng}@nlpr.ia.ac.cn
[4] Nanjing University of Information Science and Technology, Nanjing, China

Abstract. Training Binarized Neural Networks (BNNs) is challenging due to the discreteness. In order to efficiently optimize BNNs through backward propagations, real-valued auxiliary variables are commonly used to accumulate gradient updates. Those auxiliary variables are then directly quantized to binary weights in the forward pass, which brings about large quantization errors. In this paper, by introducing an appropriate proxy matrix, we reduce the weights quantization error while circumventing explicit binary regularizations on the full-precision auxiliary variables. Specifically, we regard pre-binarization weights as a linear combination of the basis vectors. The matrix composed of basis vectors is referred to as the proxy matrix, and auxiliary variables serve as the coefficients of this linear combination. We are the first to empirically identify and study the effectiveness of learning both basis and coefficients to construct the pre-binarization weights. This new proxy learning contributes to new leading performances on benchmark datasets.

Keywords: Binarized Neural Networks · Proxy Matrix

1 Introduction

Binary embedding is a fundamental technique in machine learning applications, such as retrival [12, 16], clustering [3, 19], matching [8, 38] and classification [9, 20]. The popular signum function quantizes data points to ± 1, which enables compact storage (*i.e.*, $32\times$ compression than floating point) and efficient bitwise operations (*i.e.*, replacing time-consuming inner-product with *xnor-popcnt*) [32]. However, $sgn(\cdot)$ is non-smooth with derivative 0 everywhere except at 0, which makes gradient-driven optimizations incapable, especially for training BNNs.

Pioneer works present constructive training algorithms according to the sense of growth of networks [30] and verify the information capacity of binary weights [21]. Variational Bayes methods [43, 44] propose to train discrete multilayer neural networks using Expectation Propagation (EP). Recent gradient-based methods with Straight-Through-Estimator (STE) show that a linear backprop

function for the non-linear activation surprisingly leads to promising results on CIFAR-10 [9,20,49]. XNOR-Net [37] further introduces a scaling factor to relax the binary constraint and show notable improvements on ImageNet dataset. Regardless of their differences, a real-valued auxiliary variable is commonly used to accumulate gradient updates and then binarized to ± 1 at inference time [6,20,28,37]. To minimize the weights binarization error, recent BNNs impose explicit binary regularizations on the auxiliary variables that lead to the bimodal distribution [10,13,14,45]. Though bimodality that encourages auxiliary variables to be around binary values may facilitate binarization intuitively, it can be hard to change positive auxiliary variables to negative by small gradient steps and vice versa (Note that large gradient steps can be risky for BNNs training since there are no accurate gradients for binary weights but approximations).

In this paper, we try to reduce weights quantization errors while avoiding the explicit constraint that forces the full-precision auxiliary variables to be around ± 1. To this end, we investigate the following question: is there a latent parameter space which can serve our goal, to bridge full-precision auxiliary variables and binary weights? We introduce proxy matrix R as a basis of the latent parameter space. Every filter before binarization can be written as a linear combination of basis vectors. The coefficients of this linear combination are referred to as the auxiliary variables. Since the basis can be the key component in proxy learning, we conduct empirical studies on the construction of R, based on the view of minimizing both weights quantization errors and the global cost function. It is shown that a well-designed proxy matrix leads to smooth optimization landscapes with superior performances. Exhaustive experiments show that our proxy learning strategy notably outperforms the state-of-the-art on ImageNet dataset.

2 Related works

Binarized neural network has been a long-standing topic in machine learning community [32,33]. Due to its high memory and computing efficiency, it becomes an ideal solution to the deployment of computation-intensive deep convolutional neural networks on low-power devices [4,51]. Previous literatures prove that the manual-designed backpropagation of binarization/ternarization still performs well on small datasets, not only for weights compression but activations quantization [9,20,26,49]. DoReFa [52] further presents low-bit weights, activations and gradients to accelerate both training and inference on customized devices.

To narrow the gap between BNNs and full-precision networks on the challenging ImageNet, XNOR-Net [37] proposes scaling factors for both weights and activation functions to minimize the quantization error. The following works further develop various regularization functions that encourage training weights around binary values [10,13,14] and controls the range of activations [11]. In light of the success of scaling factors, XNOR++ [6] improves the performances by learned both spatial and channel-wise scaling factors. To compensate for the information loss of binarization, Bi-Real [28] proposes double residual connections with full-precision downsampling layers and [6] replaces ReLU by PReLU. Due to

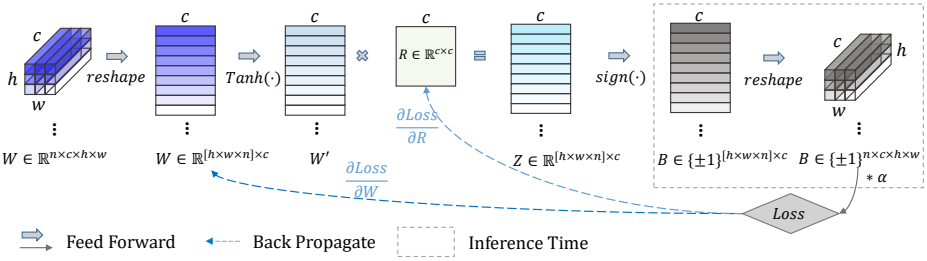


Fig. 1: Overview of the proxy learning for 3×3 binary weights

the gradient mismatch, [10, 28, 48] formulate quantization forward/backward as differentiable non-linear mapping functions. More recently, probabilistic training methods [35, 41] circumvent the need to approximate the gradient of $sign()$ by sampling from the weight distribution. Since BNN training is not well-founded, there are still tremendous efforts on the study of BNNs' optimizations [1,5,17,29] and how to explain the effectiveness of BNNs [2]. All those methods pave the way for a better understanding of binarized neural networks.

3 Methodology

3.1 Formulation

We quickly revisit the popular gradient-based method proposed in BinaryConnect [9], which maintains real-valued latent variables W for gradient updates. In the forward pass, W are binarized to ± 1 by

$$W_b = sgn(W) \tag{1}$$

to perform binary convolutions $W_b \otimes sgn(X)$, where X is the input feature map.

Given a basis R of the latent parameter space, we decompose the previous W into R and coordinates (or components) W'. Thus, we present a new pattern of learning binary weights

$$sgn(Z) = sgn(W'R), \quad W' = \phi(W) \tag{2}$$

where $W \in \mathbb{R}^{[h \times w \times n] \times c}$, $R \in \mathbb{R}^{c \times c}$ [1] and $\phi(\cdot)$ is a nonlinear mapping. As illustrated in Figure 1, during gradient descent ProxyBNN learns coordinate representations W' and updates the manual-designed basis R simultaneously. For binary activations, we assume that semantic information mainly distributed along channel dimension (*i.e.,* different channels may respond to different categories). Hence, we split each filter in spatial dimension (*i.e.,* "reshape" in Fig.1). In this

[1] h, w, n and c are kernel height, width, kernel number and input channel number, respectively. For 1×1 convolutions and FC layers, $[h \times w \times n] \times c$ degrades into $n \times c$.

way, every column of Z corresponding to the same input channel is constructed by the same basis vector $R_i \in \mathbb{R}^c$. Inspired by the common $[-1, 1]$-clip in BNNs [9, 20], we introduce the hyperbolic tangent as the activation function ϕ to cancel the gradients when W are too large. Note that W and R work as high-precision temporal variables. The extra computing and storage cost of the basis and coordinates only exist during training. At inference time, we utilize the well-trained B which is the same as previous BNNs.

3.2 Proxy learning procedure

To optimize the global objective of deep neural networks with binary constraints, we formulate the n layers BNN training to a constrained optimization problem

$$\min_{Z} \ \ell(Z), \quad s.t.\ Z_i = \alpha_i B_i,\ B_i \in \{+1, -1\}^{[h\times w]\times c}, \quad i = 1, \cdots, n \tag{3}$$

where $\alpha_i \in \mathbb{R}$ is a real-valued scaling factor to relax the binary constraint on Z_i [37] and $\ell(\cdot)$ is cross-entropy loss. Note that we introduce α_i and B_i as independent variables, which will be used in binary convolutions after training. If the first equation constraint is brought to the objective via a regularization parameter γ, we show that the resulting form can be solved by updating B_i, α_i and Z_i iteratively,

$$\mathcal{L}_\gamma = \min_{\alpha,Z,B} \ \ell(\psi(Z)) + \gamma \sum_{i=1}^{n} ||Z_i - \alpha_i B_i||_F^2, \quad s.t.\ B_i \in \{+1, -1\}^{[h\times w]\times c}, \tag{4}$$

where $\psi(\cdot)$ is a binary mapping that relaxes Z to $\mathbb{R}^{[h\times w\times n]\times c}$ and guarantees binary weights in the forward pass.

Fix Z_i, α_i, update B_i. In this step, we treat Z_i and α_i as constants and update B_i to minimize $\mathcal{L}_\gamma$. Since B_i only exists in the second term, we have

$$B_i^{t+1} = \arg\min ||Z_i^t - \alpha_i^t B_i||_F^2 = \arg\max \ tr(\alpha_i^t B_i^T Z_i^t) \tag{5}$$

where $tr(\alpha B^T Z) = \sum_{m=1}^{c} \sum_{n=1}^{h\cdot w} = \alpha B_{n,m} Z_{n,m}$. Given the binary constraint on B_i, the solution is simply $B_i^{t+1} = sgn(\alpha_i^t Z_i^t)$.

Fix Z_i, B_i, update α_i. Here we use the updated B_i^{t+1} and minimize $\mathcal{L}_\gamma$ in terms of α_i. Since Z_i^t and B_i^{t+1} are fixed in this step, problem (4) becomes independent subtasks

$$\min_{\alpha_i} ||Z_i^t - \alpha_i B_i^{t+1}||_F^2 = \min_{\alpha_i} \ (hwc)\alpha_i^2 - 2tr({B_i^{t+1}}^T Z_i^t)\alpha_i + const. \tag{6}$$

Note that α_i is a full-precision scalar and (6) is quadratic, the optimum can be easily obtained as $\alpha_i^{t+1} = \frac{tr({B_i^{t+1}}^T Z_i^t)}{hwc}$.

Fix α_i, B_i, update Z_i. To update the latent variable Z_i, we perform a gradient descent step since the objective function $\ell(\cdot)$ for BNN is differentiable and the

second term in (4) is a quadratic regularization term, which is differentiable and convex. Following the rule of SGD, the derivative of Z_i^t is calculated as follow

$$\frac{\partial \mathcal{L}_\gamma}{\partial Z_i^t} = \frac{\partial \ell}{\partial \psi(Z_i^t)} \frac{\partial \psi(Z_i^t)}{\partial Z_i^t} + 2\gamma(Z_i^t - \alpha_i^{t+1} B_i^{t+1}). \tag{7}$$

Given the optimal solution of α_i and B_i at each step, we obtain the binary mapping $\psi(Z_i) = \frac{||Z_i||_1}{c \times h \times w} sgn(Z_i)$ in vector form (*i.e.,* $Z_i \in \mathbb{R}^{[h \times w \times c] \times 1}$). Then, the gradient with respect to the k-th element in Z_i is defined as [2]

$$\frac{\partial \ell}{\partial Z_{i,k}} := \frac{sgn(Z_{i,k})}{h \cdot w \cdot c} \sum_{j=1}^{hwc} \frac{\partial \ell}{\partial \psi(Z_i)_j} sgn(Z_{i,j}) + \frac{\partial \ell}{\partial \psi(Z_i)_k}. \tag{8}$$

Combining Eq.(7) and Eq.(8), we obtain the derivative to W', R as

$$\frac{\partial \mathcal{L}_\gamma}{\partial R} = \frac{\partial \mathcal{L}_\gamma}{\partial Z}^T W', \quad \frac{\partial \mathcal{L}_\gamma}{\partial W'} = \frac{\partial \mathcal{L}_\gamma}{\partial Z} R^T. \tag{9}$$

Following the standard gradient update step in [22], $W^{t+1} \leftarrow W^t - \beta_1 \nabla_W \mathcal{L}_\gamma$ and $R^{t+1} \leftarrow R^t - \beta_2 \nabla_R \mathcal{L}_\gamma$ where β_1 and β_2 are the learning rates, we have the updated $Z^{t+1} = \phi(W^{t+1}) R^{t+1}$.

3.3 The construction of basis

Although the basis R can be trained end-to-end as shown in the previous section, we empirically prove that the construction of the initial basis matters in ProxyBNN training.

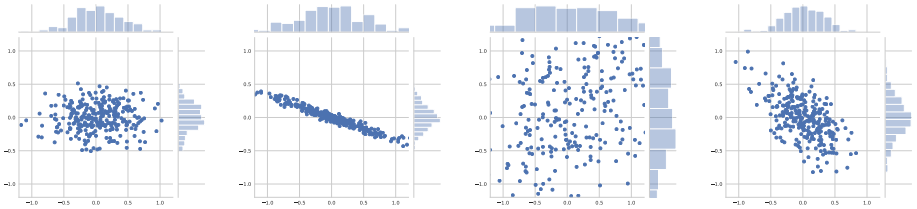
Random matrix. The most intuitive choice is a random initialization where every element $R_{i,j} \sim \mathcal{N}(0, 1)$. We include it as a baseline scheme to conduct fair comparisons.

Minimizing Square Error (MSE) matrix. In light of the empirical success of minimizing weights quantization error [10,13,14,24,37], we consider the following square object

$$\min_R \ ||W'R - sgn(W'R)||_F^2. \tag{10}$$

Beginning with the identity matrix initialization of R, we adopt an iterative optimization procedure to find a local minimum of (10). In each iteration, $W'R^t$ is first assigned to the binary codewords, and then R^{t+1} is updated to minimize the square error, *i.e.*, calculating the Moore-Penrose inverse of W' then multiplied by $sgn(W'R^t)$. Since the pseudo-inverse relies on Singular Value Decomposition (SVD), which is time-consuming for large matrix, we conduct MSE construction only once and notice no accuracy improvement (even result in worse performance) with more re-construction during training.

[2] Further details in appendix 1.



(a) W' original distribution (b) Random matrix (c) MSE matrix (d) Orthogonal matrix

Fig. 2: Toy examples of the effects of different proxy matrices. (a) shows the original distribution of W'. (b-d) illustrate the distributions of $W'R$, *i.e., Z*

Orthogonal matrix. The main idea of introducing orthogonal matrix is simply that: *similar* coordinates $w'_i, w'_j \in \mathbb{R}^{1\times c}$ may correspond to *similar* representations $z_i, z_j \in \mathbb{R}^{1\times c}$ in Euclidean space, given $z_i = w'_i R$. That is, we try to preserve the similarity relationship (locality structure) between coordinates while minimizing the quantization error. In this case, an orthogonal matrix R with $||w'_i - w'_j||_2 = ||w'_i R - w'_j R||_2$ becomes an ideal solution. Then, we reformulate problem (10) as

$$\min_{R} \ ||W'R - sgn(W'R)||_F^2, \quad s.t.\ R^T R = I. \tag{11}$$

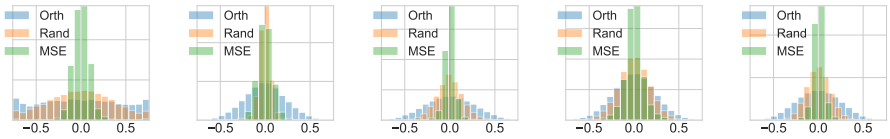
The rows of coordinate matrix $W' \in \mathbb{R}^{[h\times w\times n]\times c}$ can be seen as a set of $h \times w \times n$ data points $\{w'_1, w'_2, \cdots, w'_{h\cdot w\cdot n}\}, w'_i \in \mathbb{R}^{1\times c}$, and (11) forms the classical hashing problem. Here we use ITQ proposed in [12] for solving hashing codes to obtain the optimal R. The alternating update is similar to MSE. We first binarize $W'R$ in each step, then the objective function corresponds to the classic Orthogonal Procrustes problem [40],

$$U\Sigma V^T = \text{svd}(sgn(W'R)^T W'R), \quad R = VU^T. \tag{12}$$

Before alternating optimization, we use a random orthogonal matrix to initialize R and train 10 epochs to warm up W'. We only conduct the construction once and then update R with small gradient steps.

3.4 The effect of proxy learning

Toy example To better understand the proposed proxy learning, we first show a 2D toy example then analyze the experimental phenomenon in real networks. As shown in Figure (2b,2c), both random matrix and MSE matrix change the original data structure, especially MSE minimizes quantization errors at the cost of ruining the 2-dimensional Gaussian distribution, which approximates uniform distributions. Figure 2d shows the orthogonal matrix serves as a similarity-preserving rotation, which not only quantizes weights with small errors but maintains the structure of W'.



(a) Layer2 (b) Layer8 (c) Layer9 (d) Layer16 (e) Layer21

Fig. 3: Histograms of W' of WRN22 on CIFAR-100 (best viewed in color)

Besides, the variance of W' in each direction is different. Directly quantizing both low-variance directions and high-variance directions (with more information) to 1-bit can be suboptimal. An orthogonal matrix balances the variance of different directions (*e.g.,* different channels in real networks), which facilitates the binary encoding.

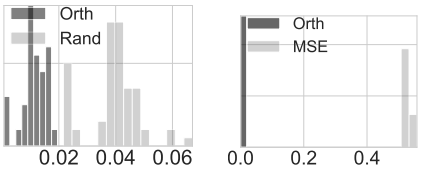


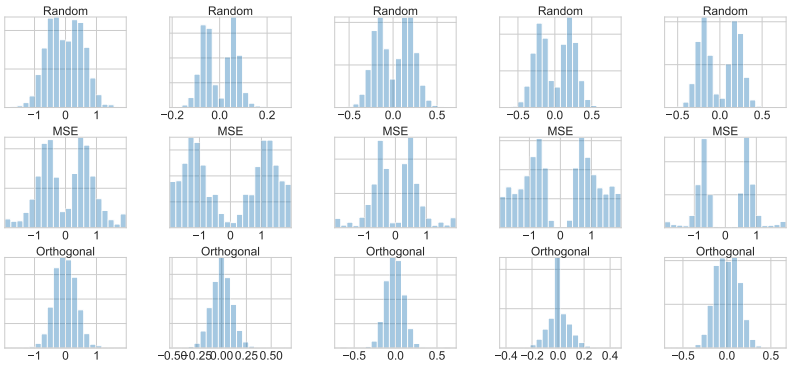
Fig. 4: Histograms of the variance of $Z \in \mathbb{R}^{h \cdot w \cdot n \cdot c}$ in the channel dimension[3]

Similar phenomena exist in practical WRN22 network. As shown in Figure 4, MSE matrix leads to the largest variance in the channel dimension among three candidates, which is consistent with Fig.2. For the random matrix, it has a wider distribution interval of variances than the orthogonal matrix, which reflects imbalanced variances across different channels, as shown in Fig.2b (*e.g.,* high variance in x-dimension and low variance in y-dimension).

Weights distribution To clearly verify the effectiveness of the proxy matrix R, we visualize the distributions of W' and $W'R$. Figure 3 illustrates that all schemes' W' are approximate Gaussian distributions similar to weights in full-precision counterparts. We further demonstrate Z in Figure 5. The baseline random matrix (*i.e.,* the first row) illustrates a bimodal distribution, which is a sensible result for pre-binarization weights to minimize quantization error. Since MSE matrix is based on $\min ||W'R - sgn(W'R)||_F$, the initial MSE basis naturally makes two peaks move towards ± 1, as shown in the second row of Figure 5. However, it seems counterintuitive, the orthogonal scheme still generates a unimodal distribution. Here is the question: *Does either the unimodal distribution or the bimodal distribution contribute to "accurate" binary networks?*

Quantization error *v.s.* classification error Table 1 details the trade-off between layer-wise quantization error and the final accuracy. Here we define the

[3] To be specific, we compute the variance of $Z_i \in \mathbb{R}^{h \cdot w \cdot n}$, $i = 1, \cdots, c$ then visualize the distribution of c samples. The more concentrated distribution indicates the more balanced variance.



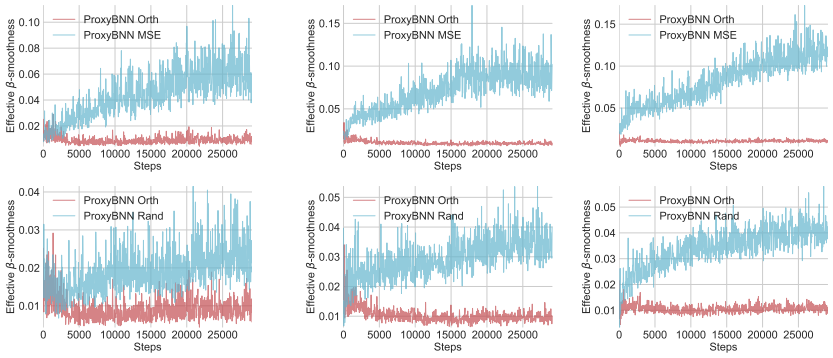
(a) Layer2 (b) Layer8 (c) Layer9 (d) Layer16 (e) Layer21

Fig. 5: Histograms of Z of WRN22 on CIFAR-100, *i.e.,* the distributions of $W'R$

Table 1: WRN-22 layer-wise weights quantization error and final accuracy. "Average" refers to the mean quantization error, averaged across all elements

Layer	ProxyBNN Orthogonal	ProxyBNN Random	ProxyBNN MSE	Bi-Real [28]
Conv2	**0.0473**	0.0834	0.0695	0.1797
Conv3	0.0108	0.0248	**0.0072**	0.1022
Conv5	0.0081	0.0139	**0.0021**	0.0664
Conv6	0.0071	0.0114	**0.0014**	0.0086
Conv7	0.0043	0.0031	**0.0011**	0.0003
Conv8	0.0043	**0.0006**	0.0009	0.0001
Conv9	**0.0043**	0.0070	0.0073	0.0670
Conv10	0.0065	0.0036	**0.0029**	0.0390
Conv12	0.0039	**0.0007**	0.0018	0.0029
Conv13	0.0023	**0.0005**	0.0010	0.0002
Conv14	0.0014	**0.0004**	0.0007	0.0001
Conv15	0.0017	**0.0003**	0.0017	0.0001
Conv16	**0.0031**	0.0089	0.0079	0.0397
Conv17	0.0068	0.0076	**0.0037**	0.0374
Conv19	0.0060	0.0069	**0.0035**	0.0373
Conv20	0.0057	0.0068	**0.0029**	0.0331
Conv21	0.0055	0.0058	**0.0035**	0.0277
Conv22	0.0056	0.0055	**0.0026**	0.0235
Average	0.0054	0.0061	**0.0035**	0.0294
Acc. (%)	**71.61**	69.10	59.32	69.73

quantization error as: $\mathcal{Q}(Z, \alpha, B) = \frac{1}{h \cdot w \cdot n \cdot c} \sum_{i=1}^{n} ||Z_i - \alpha_i B_i||_F^2$. All proxy learning schemes obtain smaller average quantization errors compared with baseline Bi-Real-Net [28]. To be specific, ProxyBNN minimizes the binarization loss in the first and last few layers, which may facilitate feature extraction and semantic



(a) Layer8 (b) Layer16 (c) Layer21

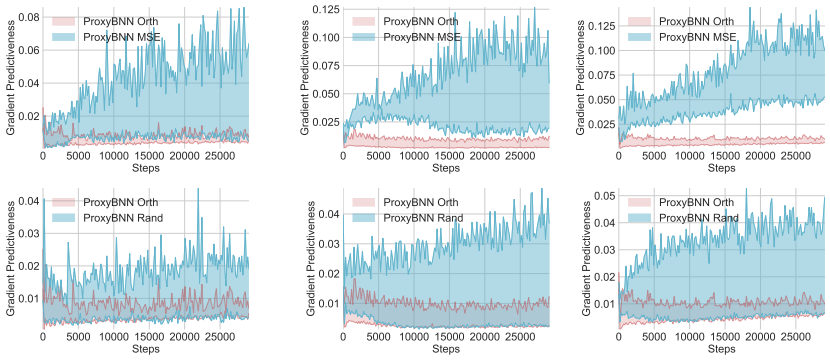
Fig. 6: Analysis of the "effective" β-smoothness [39] of WRN22 network. For a layer we measure the maximum ℓ_2-norm difference in gradient. The lower the values indicate the smoother loss landscape (best viewed in color)

analysis. Note that MSE focuses on how to quantize weights locally, which generates over 8× smaller average loss than Bi-Real, yet results in poor performance. The orthogonal scheme presents a better trade-off between weights binarization loss and the global cost function, and achieves the highest performance. It is shown that unimodal weights distributions (*i.e.*, the third row in Fig.5) can be another group of solutions to minimizing quantization error, when jointly optimized with cross-entropy loss $\ell(\cdot)$.

Optimization landscape If pre-binarization variables are close to zero, a small gradient step can change binary weights from positive to negative and vice versa, which may make the training easier. Motivated by this hypothesis, we analyze the optimization landscape of different bases and observe the superiority of the orthogonal scheme. Following [39], we measure the stability and smoothness of the landscape by Lipschitzness and "effective" β-smoothness of the loss function. As shown in Figure 6, we observe consistent differences between these schemes. The improved Lipschitzness encourages us to take a step in the direction of a computed gradient, which provides a fairly accurate estimate of the real gradient [39]. Figure 7 also demonstrates the effect of different bases on the stability/Lipschitzness of the gradients. No matter how weights quantization loss changes (Conv8/16/21 correspond to three cases in Table 1), the orthogonal scheme still outperforms other candidates.

4 Experiments

To verify the effectiveness of the proposed approach, in this section, we introduce three benchmark datasets: CIFAR-10, CIFAR-100, and ImageNet. We com-



(a) Layer8 (b) Layer16 (c) Layer21

Fig. 7: Analysis of the gradient predictiveness [39] of WRN22 network. The shaded region corresponds to the variation in ℓ_2-norm changes in gradient over the distance. The thinner shade in plots show the smoother loss landscape and thus less training difficulty (best viewed in color)

prehensively evaluate our method on the mainstream deep CNN architectures, including AlexNet [23], VGG [42], ResNet [15] and Wide ResNet [50].

4.1 Experimental setup

Network structure Since modified network structures can be the game-changer for training BNNs, we follow the same settings as prior works to make fair comparisons. For AlexNet, we use the same architecture from XNOR-Net [37] where batch normalization layers are added before activations and LRN layers are omitted. ResNet-18/34 refer to the original structure introduced in [15], unless specified. In binary weights experiments, we simply replace full-precision convolution layers with binary weights counterparts without any bells and whistles. When both activations and weights are quantized to 1-bit (including 1×1 downsample layers), we use batch-normalization before each activation function [10,37]. The modified ResNet/WRN [13, 14, 27, 28] consist of double skip connections [28], PReLU activations [7] and real-valued downsampling layers [28]. The operations are reordered as Batch-Normalization $\rightarrow$ Binarization $\rightarrow$ Binary-Convolution $\rightarrow$ Activation, as proposed in XNOR-Net [37]. VGG9 is a VGG-like structure with six convolutional layers and three fully-connected layers, first described in BinaryConnect [9]. We use the same modification as [37,46]. As in almost all previous works, the first and last layers in all experiments are kept real.

Activation binarization There have been tremendous efforts on exploiting binary activations [10, 28, 37, 46, 48]. To verify the robustness of ProxyBNN, we consider two simple settings in our experiments: the signum function proposed in BinaryNet [20] and $round(clip(x))$ introduced in DoReFa [52]. We conduct

Table 2: Performances of ProxyBNN trained with different bases. Top-1 accuracies on benchmark datasets are reported (single stage, trained from scratch)

Model	#Param.	Dataset	Orthogonal	Random	MSE
ResNet18	2.80M	Cifar10	**91.87** (±0.36)	88.36 (±0.49)	67.88 (±0.38)
	2.82M	Cifar100	**67.17** (±0.73)	53.58 (±1.08)	30.69 (±0.84)
WRN22	4.30M	Cifar10	**92.96** (±0.11)	91.24 (±0.22)	86.77 (±0.72)
	4.33M	Cifar100	**71.57** (±0.14)	68.93 (±0.29)	58.00 (±0.57)
ResNet-18	11.70M	ImageNet	**58.7**	53.7	36.8

Table 3: Error rates (%) on CIFAR-100 using WRN22

(a) Ablation studies on penalty factor γ

γ	Error (%)
0.001	30.28 (±0.11)
0.0001	28.51 (±0.10)
$1e^{-5}$	**28.43** (±0.14)
$1e^{-6}$	29.94 (±0.50)
$1e^{-7}$	31.33 (±0.47)

(b) Effect of using different initial learning rates for R

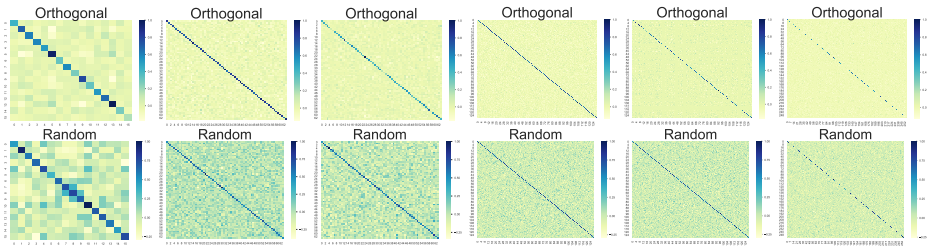
init. lr_R	Error (%)
lr_w	30.77 (±0.29)
$lr_w \times 0.1$	**28.43** (±0.14)
$lr_w \times 0.01$	30.35 (±0.05)
$lr_w \times 0.001$	31.44 (±0.30)

the first setting in CIFAR experiments then we apply the second technique to the ImageNet networks.

Ablation study In this section, we first evaluate the effects of the penalty weight γ and different learning rates for the proxy matrix. Table 3a indicates that a proper γ matters in the balance between cross-entropy loss and the penalty term. We also observe that the basis should be updated a little slower than the coordinates, as shown in Table 3b. To further verify the superiority of the orthogonal scheme, we evaluate different bases on benchmarks (Table 2). The performance gap is consistent with that in Table 1. Besides, Figure 8 shows that the property of the orthogonal basis roughly remains after training, *i.e.,* $R_i^T R_j \approx 0 \ \forall i \neq j$ (for clarity, we normalize the max value to 1). Based on Table 3, we apply the best settings to the following experiments without finetuning.

4.2 Results

CIFAR-10/100 The CIFAR-10/100 dataset consist of 50,000 train images and a test set of 10,000 across 10/100 classes. Unless specified, the images are padded by 4 pixels on each side then randomly cropped to 32×32 [13, 14, 27]. We use a batch size of 128 for training, optimized by Adam [22] with cosine scheduler. The initial learning rate for W is set to 0.005 and the weight decay is $1e^{-6}$ (same for both R and W). All networks are trained for 310 epochs.



(a) Layer2 (b) Layer3 (c) Layer9 (d) Layer10 (e) Layer16 (f) Layer17

Fig. 8: Visualizations of $R^T R$, where R is the learned proxy matrix

Table 4: Test accuracies on CIFAR-10/100, comparison with different 1-bit methods. † indicates modified architectures [28] (details in section 4.1). "MS" refers to the multi-stage training strategy, *e.g.*, binarizing the activations first, and then using the model as initialization to train fully binarized networks. "Center" means using center loss [47] during stage-2. "FP32" is the full-precision baseline. "Aug" indicates 32×32 random cropping with 4 pixels padding

Model	Kernel-Stage	Method	Cifar10 (%)	Cifar100 (%)	MS	Center	Aug
ResNet18†	16-16-32-64	PCNN [13]	78.93	41.41	✓	–	✓
		GBCN [27]	81.22	47.96	✓	✓	✓
		ProxyBNN	**84.53**	**52.07**	–	–	✓
		FP32	90.77	65.15	–	–	✓
WRN22†	64-64-128-256	PCNN [13]	91.37	69.98	✓	–	✓
		GBCN [27]	92.72	**71.85**	✓	✓	✓
		BONN [14]	92.36	–	✓	✓	✓
		ProxyBNN	**92.96**	71.57	–	–	✓
		FP32	95.75	77.34	–	–	✓
VGG9	128-256-512	BNN [20]	89.9	–	–	–	–
		XNOR [37]	89.8	–	–	–	–
		SiBNN [46]	90.2	–	–	–	–
		ProxyBNN	**90.5**	63.23	–	–	–
		FP32	91.7	67.01	–	–	–
ResNet18†	32-64-128-256	PCNN [13]	87.76	60.29	✓	–	✓
		GBCN [27]	87.69	62.01	✓	✓	✓
		ProxyBNN	**91.87**	**67.17**	–	–	✓
		FP32	93.88	72.51	–	–	✓

We compare our results with prior state-of-the-arts, as shown in Table 4. Both over-parameterized architectures, such as VGG/WRN with the kernel stage of 64-64-128-256, and compact ResNet-18 are considered. Our method in the worst case is still competitive with recent works (results reported in the original papers), without other techniques.

Table 5: Comparison with state-of-the-art methods on ResNets. "MS" refers to the multi-stage training strategy, *e.g.*, binarizing the activations first, and then using the model as initialization to train fully binarized networks. For binary weights experiments, "MS" is fine-tuning from full-precision weights. † indicates indicates modified architectures [28] (details in section 4.1)

Model	Method	Weight	Activation	Top-1 (%)	Top-5 (%)	MS
ResNet-18	XNOR [37]	1	1	51.2	73.2	–
	BNN+ [10]	1	1	53.0	72.6	✓
	QNet [48]	1	1	53.6	75.3	✓
	XNOR++ [6]	1	1	57.1	79.9	–
	ProxyBNN	1	1	**58.7**	**81.2**	–
	BWN [37]	1	32	60.8	83.0	–
	BWHN [18]	1	32	64.3	85.9	✓
	ADMM [24]	1	32	64.8	86.2	–
	IR-Net [36]	1	32	66.5	86.8	–
	ProxyBNN	1	32	**67.3**	**87.2**	–
	FP32	32	32	69.3	89.2	–
ResNet-18†	Bi-Real [28]	1	1	56.4	79.5	✓
	PCNN [13]	1	1	57.3	80.0	✓
	GBCN [27]	1	1	57.8	80.9	✓
	IR-Net [36]	1	1	58.1	80.0	–
	BONN [14]	1	1	59.3	81.6	✓
	SiBNN [46]	1	1	59.7	81.8	–
	ProxyBNN	1	1	63.3	84.3	–
	ProxyBNN	1	1	**63.7**	**84.8**	✓
	PCNN [13]	1	32	63.5	85.1	–
	ProxyBNN	1	32	**67.7**	**87.7**	–
	FP32	32	32	68.5	88.3	–
ResNet-34	ProxyBNN	1	32	**70.7**	**89.6**	–
	FP32	32	32	73.3	91.3	–
ResNet-34†	ABC [25]	1	1	52.4	76.5	–
	WRPN [31]	1	1	60.5	–	–
	Bi-Real [28]	1	1	62.2	83.9	✓
	IR-Net [36]	1	1	62.9	84.1	–
	SiBNN [46]	1	1	63.3	84.4	–
	ProxyBNN	1	1	**66.3**	**86.5**	–
	FP32	32	32	70.4	89.3	–

ImageNet ImageNet (ILSVRC2012) is one of the most challenging image classification benchmarks with over 1.2 million training images and 50K validation images, that cover 1000 object classes. As in [13, 14, 27, 28], we conduct the standard PyTorch [34] data preprocessing for both training and inference, *i.e.,* random resized 224×224 (227×227 for AlexNet) crop with the standard horizontal flip. We follow the settings in CIFAR experiments, except that the initial learning rate is set to 0.001 and the training time is 110 epochs.

For binary weights and further activation binarization, we compare the proposed algorithm with the state-of-the-art approaches. Table 5 shows the per-

Table 6: Comparison with state-of-the-art methods on AlexNet. "MS" refers to the multi-stage training strategy. For binary weights experiments, "MS" is fine-tuning from full-precision weights, otherwise, training from scratch

Model	Method	Weight	Activation	Top-1 (%)	Top-5 (%)	MS
AlexNet	DoReFa [52]	1	1	43.6	–	–
	XNOR [37]	1	1	44.2	69.2	–
	RAD [11]	1	1	47.8	71.5	–
	QNet [48]	1	1	47.9	72.5	✓
	SiBNN [46]	1	1	50.5	74.6	–
	ProxyBNN	1	1	**51.4**	**75.5**	–
	DoReFa [52]	1	32	53.9	76.3	–
	BWN [37]	1	32	56.8	79.4	–
	ADMM [24]	1	32	57.0	79.7	–
	QNet [48]	1	32	58.8	**81.7**	✓
	ProxyBNN	1	32	**59.3**	81.3	–
	FP32	32	32	61.8	83.5	–

formance gap between binary weights networks and full-precision counterparts have been narrowed to less than three points. The performance improvement in Table 6 is consistent with ResNet. When comparing to multi-bit methods such as 5 bases ABC-ResNet18 [25] with 85.9% Top-5 accuracy, our approach achieves 25× less computing cost, yet suffers only −1.1% accuracy loss.

5 Conclusions

In this paper, we present a new technique for training binarized neural networks, that decomposes pre-binarization weights into the basis and coordinates. We consider different construction schemes for the basis and empirically analyze the superiority of the orthogonal scheme. When jointly optimized by weights quantization error and cross-entropy loss, the orthogonal scheme preserves the unimodal distribution while minimizing the binarization error. Our experiments demonstrate that ProxyBNN has a better generalization capacity than previous methods on benchmark datasets. These results show that mainstream architectures can generally benefit from the proposed proxy learning, which enables the deployment of deep binarized neural networks on low-power devices.

Acknowledgement

This work was supported in part by National Natural Science Foundation of China (No.61972396, 61876182, 61906193), National Key Research and Development Program of China (No. 2019AAA0103402), the Strategic Priority Research Program of Chinese Academy of Science(No.XDB32050200), the Advance Research Program (No. 31511130301), and Jiangsu Frontier Technology Basic Research Project (No. BK20192004).

References

1. Alizadeh, M., Fernndez-Marqus, J., Lane, N.D., Gal, Y.: A systematic study of binary neural networks' optimisation. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=rJfUCoR5KX
2. Anderson, A.G., Berg, C.P.: The high-dimensional geometry of binary neural networks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=B1IDRdeCW
3. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun. ACM **51**(1), 117–122 (2008). https://doi.org/10.1145/1327452.1327494, http://doi.acm.org/10.1145/1327452.1327494
4. Bahou, A.A., Karunaratne, G., Andri, R., Cavigelli, L., Benini, L.: XNORBIN: A 95 top/s/w hardware accelerator for binary convolutional neural networks. In: 2018 IEEE Symposium in Low-Power and High-Speed Chips, COOL CHIPS 2018, Yokohama, Japan, April 18-20, 2018. pp. 1–3. IEEE Computer Society (2018). https://doi.org/10.1109/CoolChips.2018.8373076, https://doi.org/10.1109/CoolChips.2018.8373076
5. Bethge, J., Yang, H., Bornstein, M., Meinel, C.: Back to simplicity: How to train accurate bnns from scratch? CoRR **abs/1906.08637** (2019), http://arxiv.org/abs/1906.08637
6. Bulat, A., Tzimiropoulos, G.: Xnor-net++: Improved binary neural networks. In: British Machine Vision Conference, BMVC 2019 (2019)
7. Bulat, A., Tzimiropoulos, G., Kossaifi, J., Pantic, M.: Improved training of binary networks for human pose estimation and image recognition. CoRR **abs/1904.05868** (2019), http://arxiv.org/abs/1904.05868
8. Cheng, J., Leng, C., Wu, J., Cui, H., Lu, H.: Fast and accurate image matching with cascade hashing for 3d reconstruction. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014. pp. 1–8 (2014). https://doi.org/10.1109/CVPR.2014.8, https://doi.org/10.1109/CVPR.2014.8
9. Courbariaux, M., Bengio, Y., David, J.: Binaryconnect: Training deep neural networks with binary weights during propagations. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada. pp. 3123–3131 (2015), http://papers.nips.cc/paper/5647-binaryconnect-training-deep-neural-networks-with-binary-weights-during-propagations
10. Darabi, S., Belbahri, M., Courbariaux, M., Nia, V.P.: BNN+: improved binary network training. CoRR **abs/1812.11800** (2018), http://arxiv.org/abs/1812.11800
11. Ding, R., Chin, T., Liu, Z., Marculescu, D.: Regularizing activation distribution for training binarized deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. pp. 11408–11417 (2019)
12. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **35**(12), 2916–2929 (2013). https://doi.org/10.1109/TPAMI.2012.193, https://doi.org/10.1109/TPAMI.2012.193
13. Gu, J., Li, C., Zhang, B., Han, J., Cao, X., Liu, J., Doermann, D.S.: Projection convolutional neural networks for 1-bit cnns via discrete back propagation. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019,

The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. pp. 8344–8351 (2019). https://doi.org/10.1609/aaai.v33i01.33018344, https://doi.org/10.1609/aaai.v33i01.33018344

14. Gu, J., Zhao, J., Jiang, X., Zhang, B., Jianzhuang, L., Guo, G., Ji, R.: Bayesian optimized 1-bit cnns. In: IEEE Proceedings of the IEEE International Conference on Computer Vision ICCV 2019, Seoul, South Korea (2019)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90, https://doi.org/10.1109/CVPR.2016.90
16. He, X., Wang, P., Cheng, J.: K-nearest neighbors hashing. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. pp. 2839–2848 (2019), http://openaccess.thecvf.com/content_CVPR_2019/html/He_K-Nearest_Neighbors_Hashing_CVPR_2019_paper.html
17. Helwegen, K., Widdicombe, J., Geiger, L., Liu, Z., Cheng, K., Nusselder, R.: Latent weights do not exist: Rethinking binarized neural network optimization. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada. pp. 7531–7542 (2019), http://papers.nips.cc/paper/8971-latent-weights-do-not-exist-rethinking-binarized-neural-network-optimization
18. Hu, Q., Wang, P., Cheng, J.: From hashing to cnns: Training binary weight networks via hashing. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 3247–3254 (2018), https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16466
19. Hu, Q., Wu, J., Bai, L., Zhang, Y., Cheng, J.: Fast k-means for large scale clustering. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017. pp. 2099–2102 (2017). https://doi.org/10.1145/3132847.3133091, https://doi.org/10.1145/3132847.3133091
20. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks. In: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain. pp. 4107–4115 (2016), http://papers.nips.cc/paper/6573-binarized-neural-networks
21. Ji, C., Psaltis, D.: Capacity of two-layer feedforward neural networks with binary weights. IEEE Trans. Information Theory **44**(1), 256–268 (1998). https://doi.org/10.1109/18.651033, https://doi.org/10.1109/18.651033
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980

23. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017). https://doi.org/10.1145/3065386, http://doi.acm.org/10.1145/3065386
24. Leng, C., Dou, Z., Li, H., Zhu, S., Jin, R.: Extremely low bit neural network: Squeeze the last bit out with ADMM. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 3466–3473 (2018), https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16767
25. Lin, X., Zhao, C., Pan, W.: Towards accurate binary convolutional neural network. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA. pp. 345–353 (2017), http://papers.nips.cc/paper/6638-towards-accurate-binary-convolutional-neural-network
26. Lin, Z., Courbariaux, M., Memisevic, R., Bengio, Y.: Neural networks with few multiplications. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings (2016), http://arxiv.org/abs/1510.03009
27. Liu, C., Ding, W., Hu, Y., Zhang, B., Liu, J., Guo, G.: Gbcns: Genetic binary convolutional networks for enhancing the performance of 1-bit dcnns. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20) (February 2020)
28. Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., Cheng, K.: Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In: Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV. pp. 747–763 (2018). https://doi.org/10.1007/978-3-030-01267-0_44, https://doi.org/10.1007/978-3-030-01267-0_44
29. Martinez, B., Yang, J., Bulat, A., Tzimiropoulos, G.: Training binary neural networks with real-to-binary convolutions. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=BJg4NgBKvH
30. Mayoraz, E., Aviolat, F.: Constructive training methods for feedforward neural networks with binary weights. International journal of neural systems **7 2**, 149–66 (1995)
31. Mishra, A.K., Nurvitadhi, E., Cook, J.J., Marr, D.: WRPN: wide reduced-precision networks. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings (2018), https://openreview.net/forum?id=B1ZvaaeAZ
32. Oliveira, A.L., Sangiovanni-Vincentelli, A.L.: Learning complex boolean functions: Algorithms and applications. In: Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]. pp. 911–918 (1993), http://papers.nips.cc/paper/857-learning-complex-boolean-functions-algorithms-and-applications
33. Pagallo, G., Haussler, D.: Boolean feature discovery in empirical learning. Machine Learning **5**, 71–99 (1990). https://doi.org/10.1007/BF00115895, https://doi.org/10.1007/BF00115895
34. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In:

Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada. pp. 8024–8035 (2019), http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library
35. Peters, J.W., Genewein, T., Welling, M.: Probabilistic binary neural networks (2019), https://openreview.net/forum?id=B1fysiAqK7
36. Qin, H., Gong, R., Liu, X., Wei, Z., Yu, F., Song, J.: Ir-net: Forward and backward information retention for highly accurate binary neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle Wastington, USA (June 2020)
37. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV. pp. 525–542 (2016). https://doi.org/10.1007/978-3-319-46493-0_32, https://doi.org/10.1007/978-3-319-46493-0_32
38. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.R.: ORB: an efficient alternative to SIFT or SURF. In: IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011. pp. 2564–2571 (2011). https://doi.org/10.1109/ICCV.2011.6126544, https://doi.org/10.1109/ICCV.2011.6126544
39. Santurkar, S., Tsipras, D., Ilyas, A., Madry, A.: How does batch normalization help optimization? In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada. pp. 2488–2498 (2018), http://papers.nips.cc/paper/7515-how-does-batch-normalization-help-optimization
40. Schonemann, P.H.: A generalized solution of the orthogonal procrustes problem. Psychometrika **31**(1), 1–10 (1966)
41. Shayer, O., Levi, D., Fetaya, E.: Learning discrete weights using the local reparameterization trick. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net (2018), https://openreview.net/forum?id=BySRH6CpW
42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1409.1556
43. Soudry, D., Hubara, I., Meir, R.: Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In: Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada. pp. 963–971 (2014), http://papers.nips.cc/paper/5269-expectation-backpropagation-parameter-free-training-of-multilayer-neural-networks-with-continuous-or-discrete-weights
44. Soudry, D., Meir, R.: Mean field bayes backpropagation: scalable training of multilayer neural networks with binary weights (2013)
45. Tang, W., Hua, G., Wang, L.: How to train a compact binary neural network with high accuracy? In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. pp. 2625–2631 (2017), http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14619

46. Wang, P., He, X., Li, G., Zhao, T., Cheng, J.: Sparsity-inducing binarized neural networks. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20) (February 2020)
47. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VII. pp. 499–515 (2016). https://doi.org/10.1007/978-3-319-46478-7_31, https://doi.org/10.1007/978-3-319-46478-7_31
48. Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., Hua, X.s.: Quantization networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach CA, USA (June 2019)
49. Yazdani, M.: Linear backprop in non-linear networks. In: Compact Deep Neural Network Representation with Industrial Applications Workshop, Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada (2018)
50. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016 (2016), http://www.bmva.org/bmvc/2016/papers/paper087/index.html
51. Zhao, T., He, X., Cheng, J., Hu, J.: Bitstream: Efficient computing architecture for real-time low-power inference of binary neural networks on cpus. In: Boll, S., Lee, K.M., Luo, J., Zhu, W., Byun, H., Chen, C.W., Lienhart, R., Mei, T. (eds.) 2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018. pp. 1545–1552. ACM (2018). https://doi.org/10.1145/3240508.3240673, https://doi.org/10.1145/3240508.3240673
52. Zhou, S., Ni, Z., Zhou, X., Wen, H., Wu, Y., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. CoRR **abs/1606.06160** (2016), http://arxiv.org/abs/1606.06160