Binarizing MobileNet via Evolution-based Searching

Hai Phan[1] * Zechun Liu[1,3] * Dang Huynh[2] Marios Savvides[1] Kwang-Ting Cheng[3] Zhiqiang Shen[1]
[1]Carnegie Mellon University [2]Axon Enterprise [3] Hong Kong University of Science and Technology
{haithanp,marioss,zhiqians}@andrew.cmu.edu dhuynh@axon.com {zliubq,timcheng}@ust.hk

Abstract

*Binary Neural Networks (BNNs), known to be one among the effectively compact network architectures, have achieved great outcomes in the visual tasks. Designing efficient binary architectures is not trivial due to the binary nature of the network. In this paper, we propose a use of evolutionary search to facilitate the construction and training scheme when binarizing MobileNet, a compact network with separable depth-wise convolution. Inspired by one-shot architecture search frameworks, we manipulate the idea of group convolution to design efficient 1-Bit Convolutional Neural Networks (CNNs), assuming an approximately optimal trade-off between computational cost and model accuracy. Our objective is to come up with a tiny yet efficient binary neural architecture by exploring the best candidates of the group convolution while optimizing the model performance in terms of complexity and latency. The approach is threefold. First, we train strong baseline binary networks with a wide range of random group combinations at each convolutional layer. This set-up gives the binary neural networks a capability of preserving essential information through layers. Second, to find a good set of hyper-parameters for group convolutions we make use of the evolutionary search which leverages the exploration of efficient 1-bit models. Lastly, these binary models are trained from scratch in a usual manner to achieve the final binary model. Various experiments on ImageNet are conducted to show that following our construction guideline, the final model achieves **60.09%** Top-1 accuracy and outperforms the state-of-the-art CI-BCNN with the same computational cost.*

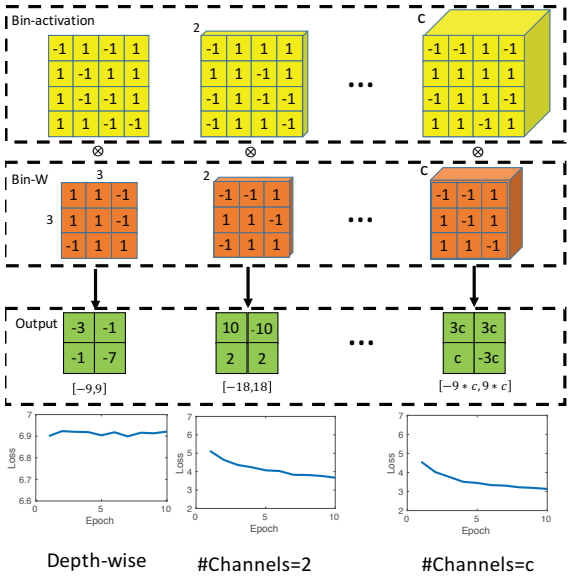

Figure 1. XNOR binary operation at depth-wise (left), group (middle), and fully (right) convolutional layers. ⊗ denotes the XNOR operator. Assuming all channels have the same value, the value range of the depth-wise convolution is constrained in $[-9, 9]$, limiting the representation capability of BNNs but fast to convolve. The full binary convolution provides a larger possible value range, enhancing the capability of BNNs but slow to compute. The group convolution balances both worlds: it maintains a sufficiently wide value range to preserve the feature representation while being efficiently light-weight. The three figures in the bottom indicate the convergence behavior of the depth-wise, group and full convolution respectively during training phase. The binary depth-wise is prone to divergence; the full convolution effectively finds a way to the local mimima and is slightly better than the group convolution which steadily converges.

1. Introduction

In the last few years, Deep Convolutional Neural Network (DCNN) for mobile platforms, which assumes certain constraints on computational capacity and battery, has been experimentally proven to be a successful approach in a wide variety of visual tasks in machine vision [20, 16, 43, 42, 52, 33]. Many compressed neural networks were proposed such as pruning [13, 12, 29, 49, 50, 31] and quantization [1, 54, 25]. Binary Neural Networks (BNNs) recently have attracted many interests and achieved significant improvements [39, 15, 32, 3, 45]. Prior works focused on binarizing large ConvNets which often contain several millions of parameters. On the other hand, compact neural network (e.g., MobileNet [16]) is among promissing network architec-

*indicates equal contribution.

2575-7075/20/$31.00 ©2020 IEEE
DOI 10.1109/CVPR42600.2020.01343

tures for binarization. The MobileNet exploits light-weight depth-wise and point-wise convolution layers to leverage the network efficiency when deploying on mobile devices. However, it is not trivial to make the depth-wise operators capable of coping with 1-bit quantization to push the network more compact.

With the depth-wise convolution, the neural network achieves low inference latency and even more optimal when being 1-bit quantized. However in such a binarization, input of the convolutional layers are channel-wise multiplied and summed. Therefore, the output values are limited within a narrow range. For instance, a binary 3×3 depth-wise filter convolving with one channel of the input yields values in $[-9, 9]$, degrading the representation capability of the binary neural networks. On the other hand, the binary vanilla convolution results in a larger output value range which allows to attain an abundant feature representation and to effectively preserve the distribution of the data samples through network layers. Figure 1 illustrates the principle of this perspective. Although being effective, the vanilla convolution comes with an expensive computational cost since the filter convolves all channels of the input tensor. Therefore, the replacement of either depth-wise or vanilla by group convolution appears to be a promising approach to compensate the trade-off between the neural network latency, feature representation capability and computational resource constraint.

Group convolution is a simple yet efficient operation used in various neural networks to optimize trainable network parameters as well as the computation. AlexNet [23], ConDenseNet [17], ResNeXt [47], etc. are among popular neural architectures exploiting the group convolution and achieving great outcomes. At its extremity appears the depth-wise convolution. Inside the depth-wise, each channel is a separate group, or in other words the number of groups is exactly the depth of the input tensor. Vanilla convolution is also a special case of group convolution where #groups $= 1$. In most of the networks having group convolutional layers, the number of groups is often homogeneous at different layers in the network. From our perspective, a heterogeneous scheme to distribute groups at different layers can help to construct an efficient and accurate neural architecture, intuitively assuming a non-homogeneous feature representation through network layers.

Aiming at leveraging the effectiveness of the heterogeneous group convolution, in this paper we propose a novel weight-sharing mechanism to explore in group search space optimally compact binary neural architectures that work efficiently and accurately. The key idea is to formulate the searching as an optimization problem that seeks to create a new genre of the compact architecture. This network is expected to be capable of performing efficiently in challenging and complicated tasks of image classification when data volume is huge and objects are diverse in types. Instead of conducting the search in a convolutional operation space with high degree of intractability as in neural architecture search (NAS) [55, 37, 48, 27], we exploit a controllable search space of group convolution in a MobileNet structure consisting of 13 layers [16], resulting in a potentially compact yet efficient binary architecture.

The main contributions of this paper are threefold:

- We introduce a novel construction of binary neural network that is one of the first studies searching for a potential architecture design via a heterogeneous combination of group convolutional layers. Our work sheds the light on a new direction for enhancing the capability of BNNs.
- We propose an adaptive weight-sharing training mechanism that automatically searches in the group space to build efficient BNNs. More importantly, our training scheme is intuitive, flexible, and straightforward to implement.
- We extensively conduct experiments to prove that following our approach, the binary neural architecture construction achieves a significant improvement factor regarding computation saving and model accuracy, therefore being able to attain state-of-the-art performance on large-scale ImageNet dataset [9].

2. Related Work

We have witnessed many research interests in binary neural networks. Courbariaux et al. [8, 7, 18] described the very first works to constrain full-precision weights in deep convolution neural networks to $\{-1, 1\}$ by utilizing XNOR-count operator and being able to accelerate the inference stage $23\times$ faster than standard convolutional operation and $3.4\times$ than cuBlas [51], an efficient GPU framework used for linear algebra computation. The work achieves high accuracy when benchmarking on popular datasets such as MNIST [24], CIFAR10 [22], SHVN [36]. XNOR-Net [39] is an interesting idea making use of scaling factors estimated from full-precision weights and achieving 44% Top-1 accuracy on ImageNet with AlexNet architecture [23]. The two most related approaches to our works are Bi-RealNet [32] and MoBiNet [38]. These binary models described a deployment of compact modules with skip connection and group convolution to enhance the capability of BNNs in terms of feature representation. The two models reach the state-of-the-art performance of 56% and 54% Top-1 accuracy on Imagenet respectively when binarizing both activations and weights. A recent work on BNNs [15] introduced Binary Optimizer to remove the dependency of binary weights from the real values, opening a new way to improve the BNNs.

To ameliorate the Binary Neural Network architecture, we adopt the methodology of Neural Architecture Search (NAS). The NAS aims at seeking to construct neural networks in an automatic instead of a manual manner. Many NAS algorithms formulate the search as an optimization problem and achieve great success [28, 5] in finding optimal architectures, assuming constraints on network latency and computational resource. In the following we focus on reviewing the neural architecture search appropriate to apply for mobile devices. Pham et al. introduced ENAS [37] considered as one of the first efficient neural architecture search approaches using cell-based search space. This network trains a super-graph from which sub-optimal paths are selected to create sharing parameters in sub-models. This mitigates the challenges when wandering in a huge exploration space by shrinking the search process parameters. There are other approaches outperforming manually designed networks. Liu et al. [28] proposed DARTS, a prominent gradient-based method that optimizes jointly one-shot models on a continuous relaxation of the search space. However because the models are assembled by a mixture from a set of operations, the performance relies heavily on the set selection. Another approach having the same flavor is ProxylessNAS [4] which adapts 1-bit neural architecture to abate GPU memory usage of one-shot models. The probability to select operation edges is updated by BinaryConnect [8].

While NAS algorithms based on reinforcement learning and evolutionary methods strictly demand prohibitive computation with thousands of GPUs [55, 27, 40, 41, 34], single-path one-shot architecture search methods are affordable over a conditional exploration space. Guo et al. [11] and Chen et al. [6] implemented the one-shot model named SPOS and DetNAS to solve image classification and object detection problem, respectively. SPOS [11] delves into a random single path at every iteration to set up a super network on which the algorithm applies an evolutionary search to seek for an optimal path for neural network formation. In the one-shot network, pre-trained output can be used to transfer to different types of task like object detection and segmentation. Our proposed method has a similar flavor in training random group convolution, assuming modifications in the neural architecture with weight sharing and searching for the optimal group combinations.

3. Our Methodology

3.1. Binary Operation

In this section, we provide some fundamental background on binary neural network. When binarizing weights and activations, a typical binary neural network uses a sign function to constrain values to either -1 or $+1$.

$$\mathbf{x}^b = \text{Sign}(\mathbf{x}) = \begin{cases} +1, & \text{if } \mathbf{x} \geqslant 0, \\ -1, & \text{otherwise} \end{cases} \tag{1}$$

where $\mathbf{x}^b$ is binarized value of x which can be network inputs or weights. Similar to float-type neural network, 1-bit weights are intentionally computed to minimize an objective function:

$$\mathbf{w}^{b*} = \arg\min_{\mathbf{w}^b} L(f_b(\mathbf{x}, \mathbf{w}^b), y) \tag{2}$$

where L is the loss function; $\mathbf{x}$, $\mathbf{w}^b$, y are inputs, binary weights, and labels respectively. Because 1-bit values degrade the neural network capability of preserving feature through layers, we apply scaling factors and backpropagation scheme mentioned in XNOR-Net [39] to tackle the training divergence issue and to enhance the binary network performance. Also, to compute gradient of non-differentiable sign function, we adapt an approximation for the derivative of the sign function with respect to the activation [32].

$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{X}^b}\frac{\partial \mathbf{X}^b}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{X}^b}\frac{\partial \text{Sign}(\mathbf{X})}{\partial \mathbf{X}} \approx \frac{\partial L}{\partial \mathbf{X}^b}\frac{\partial A(\mathbf{X})}{\partial \mathbf{X}} \tag{3}$$

$A(\cdot)$ denotes a differentiable approximation function in a piece-wise polynomial function [32], where

$$A(x) = \begin{cases} \text{-1}, & \text{if x} < -1, \\ 2x + x^2, & \text{if } -1 \leq x < 0, \\ 2x - x^2, & \text{if } 0 \leq x < 1, \\ 1, & \text{otherwise.} \end{cases}$$
$$\frac{\partial A(x)}{\partial x} = \begin{cases} 2 + 2x, & \text{if } -1 \leq x < 0, \\ 2 - 2x, & \text{if } 0 \leq x < 1, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The weights are only binarized in forward step for both training and testing stage, then we can apply binary xnor-popcount operator [35, 2] to accelerate the process. In backward step, real weights are stored to compute the derivatives and update new values.

3.2. Design And Search 1-Bit MobileNets

Binary neural network and neural architecture search are two among the most potential techniques used to construct compact yet efficient neural models. Network architecture design usually has a great impact on the performance of the binary networks. The main objective in our work is to explore efficient designs of BNNs with the hope that techniques in neural architecture search (NAS) can leverage the exploration for compact structures. However, the NAS often covers a huge search space of convolutional operators so that it is able to generate sub-optimal neural networks. This might be very difficult and costly when directly exploring in the binary operator space. To simplify the search space and to prevent the computation from exorbitant price,

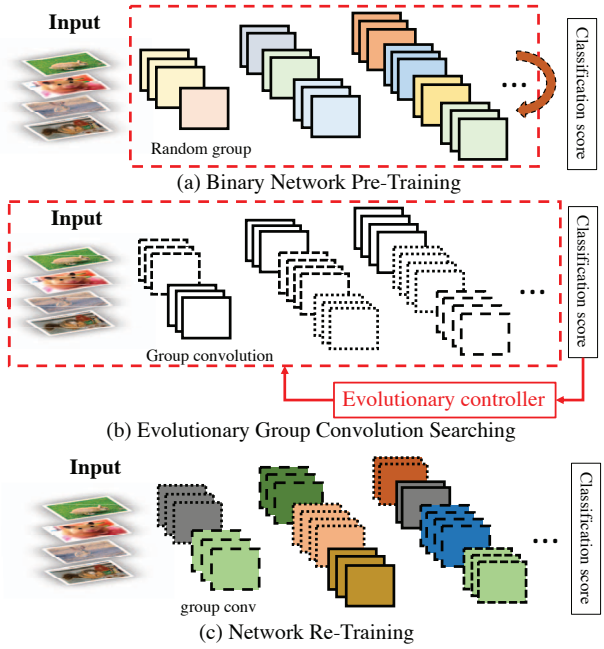

Figure 2. Framework overview of our proposed method. The process consists of three steps: train binary models with random groups (top figure, curve arrow indicates looping), apply evolutionary search to explore optimal groups based on accuracy metric, and re-train the searched group models from scratch.

we develop a novel training procedure exploiting randomized group convolution operators with weight sharing in the neural networks. With this approach, the binary models become robust thanks to the consideration of a wide variety of group combinations which fosters the group search procedure. Exploring new architecture for binary neural networks using neural architecture search can open a potential research direction to significantly improve the binary network construction. In the next sections, we discuss how to conduct and optimize the group convolution search with our proposed training pipeline.

3.2.1 Evolutionary Group Convolution Search for Binary Neural Networks

To our knowledge, evolutionary algorithms, a.k.a genetic algorithms, base on the well-known evolution of creature species in nature. Natural selection eliminates individuals unable to adapt to the environment. Additionally, survivals are kept for reproduction, crossover, and mutation. Several recent evolutionary approaches for neural architecture are proposed [30, 46]. Instead of searching for the entire network including a complete set of connections and operators as in prior works, we conduct an evolutionary search for group values at convolutional layers to explore suitable binary structure with a simple and effective network design. At each layer, group candidatures consist of all possible divisors of the input channels. In detail, we start by sampling a list of possible groups and searching on this list to find an optimal architecture by training random groups for every iteration. The first objective is to achieve an accuracy superior than a threshold. Second, in order to make the computational cost controllable, we select binary compact models satisfying certain constraint on the maximum number of FLOPs such that

$$\text{FLOP}(\mathbf{W}, \mathcal{G}) \leq \text{FLOP}_{\text{max}} \tag{5}$$

where $\mathbf{W}$ is weight and $\mathcal{G}$ is a group combination for each convolutional layer respectively. The search pipeline is presented in Algorithm 1.

Algorithm 1 Evolutionary Search for Group Convolution

Input: Candidate Group Size: $\mathcal{S}$, Top Candidates: $\mathcal{K}$, #Crossovers: $\mathcal{C}$, #Mutations: $\mathcal{M}$, Model weights: $\mathcal{W}$, and FLOP constraint: $\mathcal{F}$

Output: Optimal group combination $\mathcal{G}^*$ that yields top accuracy among the other combinations.

$\mathcal{G}^* \leftarrow$ *Sample_Candidates*$(\mathcal{S}, \mathcal{F})$ # group candidates
for i=1:maxIteration **do**
  Fitness $\leftarrow$ *Accuracy*$(\mathcal{W}, \mathcal{G}^*)$ # accuracies
  $\mathbf{K} \leftarrow$ *Select_TopK*(Fittness, $\mathcal{K}$)
  $\mathbf{C} \leftarrow$ *Crossover*$(\mathbf{K}, \mathcal{C})$
  $\mathbf{M} \leftarrow$ *Mutation*$(\mathbf{C}, \mathcal{M})$
  $\mathcal{G}^* \leftarrow \mathbf{C} \cup \mathbf{M}$
end for

3.2.2 Module Modification

MobileNets [16] with depth-wise and point-wise convolution (together known as separable depth-wise convolution [44]) are famous for its compactness and effectiveness when being used for designing a neural network. We modify the MobileNet structure to facilitate the creation of efficient binary neural networks that outperform prior state-of-the-art works regarding accuracy and memory saving. However, training a binary depth-wise convolution is not straightforward [38] because the separate channel-wise output falls into a small value range due to the nature of the computation, making the binary network impossible to converge. To overcome this issue, we propose a replacement of depth-wise convolution by group convolution to enlarge the value range of the depth-wise convolution output. More precisely, we search for groups of binary convolution operators of kernel size 3×3 and 1×1. To preserve the feature representation through binary layers while assuming a low computational cost, we maintain the full precision 1×1 convolution when perceiving a reduction in spatial dimension at a layer

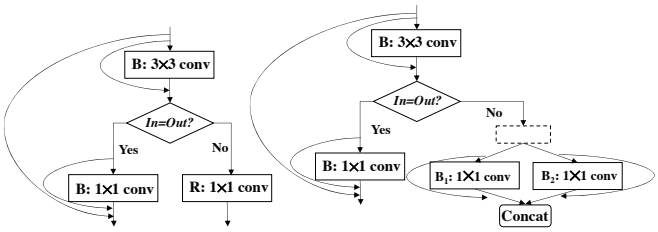

Figure 3. Illustration of network module modification.

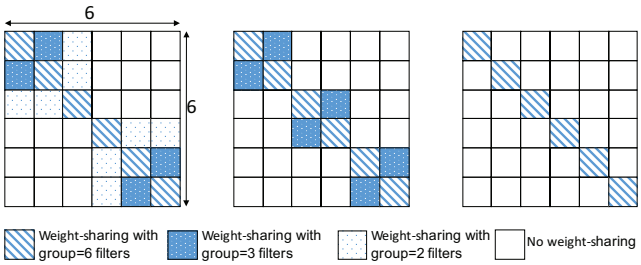

Figure 4. Illustration of a 2D weight sharing. For example, in a 6×6 filter, weights can be shared within groups of 2, 3, and 6.

output. In addition, block-wise and layer-wise skip connections are added in case of homogeneous dimension output to benefit the network training. Our proposed network module is illustrated in Figure 3. There are three principle modifications that make our modules different from the vanilla architecture of the MobileNet [16]:

- **Module 1 (M1)**: consists of a binary 3×3 group conv and a binary 1×1 full conv. A real 1×1 fully conv follows when there is a spatial dimension shrinkage. (see Figure 3 - the left figure).
- **Module 2 (M2)**: uses group convolution for real 1×1 full conv to further reduce the computational cost. The group is also searched along with the binary 3×3 group conv.
- **Module 3 (M3)**: is made up of two binary 1×1 conv layers instead of one, and then concatenate them to obtain the same dimension.

In the next section, we describe a training scheme based on randomized group through weight sharing to force the binary neural network to converge.

3.2.3 Randomized Groups via Weight-sharing

In the search stage, a fitness function (e.g., accuracy of the model) is computed to help explore optimal group combinations. However if we naively calculate accuracy of a binary model without training with data samples (i.e., images), it does not guarantee that the optimal model is able to learn the distribution of the target dataset. Therefore, to leverage important information from a given dataset for evolutionary search, we propose a method to train the binary model along with randomized group combination via weight-sharing in each training iteration. To ease the implementation, full convolutions are initialized and cropped with randomized groups in each iteration via weight-sharing. The weight-sharing is depicted in Figure 4.

3.2.4 Training Procedure

Binary neural network is an active and progressive research topic with prominent works [38, 32, 45]. Training neural networks of 1-bit weights is a difficult task because feature representation often has narrow value range which seems to be impossible to fit the target large-scale dataset for classification. XNOR-Net [39] utilizes full precision weight values to derive real value scaling factors which play crucial role in amplifying the magnitude of binary weights and activations. The optimization is formulated as follows:

$$\mathbf{X}^{b*}, \boldsymbol{\alpha} = \underset{\mathbf{X}^b \in \{-1,1\}, \boldsymbol{\alpha} > 0}{\arg\min} \left\| \mathbf{X} - \boldsymbol{\alpha} \mathbf{X}^b \right\|_2^2 \tag{6}$$

$\mathbf{X}$ can be weights or activations and $\boldsymbol{\alpha}$ are scaling factors. The optimal solution for Equation 6 is $\mathbf{X}^{b*} = \text{Sign}(\mathbf{X})$ and $\boldsymbol{\alpha} = \frac{1}{(\mathbf{X}^b)^T \mathbf{X}^b} |\mathbf{X}|_{l1}$. Bi-RealNet [32] and MoBiNet [38] make use of skip connections to enhance the performance of binary neural networks. With that flavor, we manipulate the skip connections together with scaling factors to facilitate the training procedure. Here follows the summary of our training:

- For each iteration, we train binary neural networks with random group combinations. For instance, if the network has 13 layers, the groups corresponding to these layers are randomized within possible divisors of input channels. This randomization helps 1-bit models become robust against group changes when searching.
- Evolutionary search described in Algorithm 1 is applied to seek for optimal groups. An ablation study is conducted in Section 4.2 to prove that our search approach is more efficient than arbitrary randomized groups.
- We train from scratch the final binary models with optimal group convolution. All steps run on large scale ImageNet-1k [9].

The training pseudo-code is illustrated in Algorithm 2 and visualized in Figure 2.

4. Experiments

In this section, we demonstrate the performance evaluation of our proposed method. First, we describe experi-

Algorithm 2 Overall Training BNNs

Input: Full binary neural model and inputs for evolutionary search

Output: New optimal binary neural model with new group structure.

1: $C\_ins \leftarrow Input\_Channels$
2: Initialize Binary Models $\mathcal{M}_b$
3: **for** i=1:Iteration **do**
4: $\quad \mathcal{G}_r \leftarrow Random\_Groups(C\_ins)$ # random group
5: $\quad Train\_Group(\mathcal{M}_b, \mathcal{G}_r)$
6: **end for**
7: $\mathcal{G}^* \leftarrow Search\_Group(\mathcal{M}_b)$ # search group using Algorithm 1
8: $Train\_Group(\mathcal{M}_b, \mathcal{G}^*)$

ment setups and implementation details. Second, to prove our weight-sharing group search mechanism more effective and reasonable than naively random search we compare the training performance with randomized groups in ablation studies. Third, we evaluate the search groups with uniform normal groups to investigate the fact that for each level of feature representation, the number of groups should be different. Then we compare with the state-of-the-arts to see improvement impacts of our proposed BNNs. Finally, the computation analysis are presented. All experiments including searching, training, and testing are conducted on the large-scale dataset of ImageNet2012-1k. We analyzed the results regarding three metrics: Top-1, Top-5 classification accuracy on ImageNet dataset, and number of FLOPs.

4.1. Experimental Setups

Dataset. The image dataset we used to demonstrate the effectiveness of our framework is ILSVRC2012 [9], a dataset containing 1.2M and 50K image samples for training and testing respectively. The dataset has 1000 classes. Most of the previous works such as XNOR-Net [39], Bi-RealNet [32], CI-BCNN [45], and MoBiNet [38] also used this dataset to evaluate their model performance.

Implementation details and setups. Our training pipeline consists of three main stages: train binary architecture with randomized groups, search groups for convolutional layers via evolutionary method, and train the final models with searched groups from scratch. We train on basic blocks modified from MobileNet to improve the performance of binary models, mentioned in Section 3.2.2. Each image is scaled up to 256×256. In training, images are randomly cropped to 224×224. In testing, they are centrally cropped to 224×224. When training, real-valued filters are saved in RAM to compute update values in backpropagation via Equation 3 and then are binarized in inference stage. In the first stage of the training pipeline, we used batch size of 512

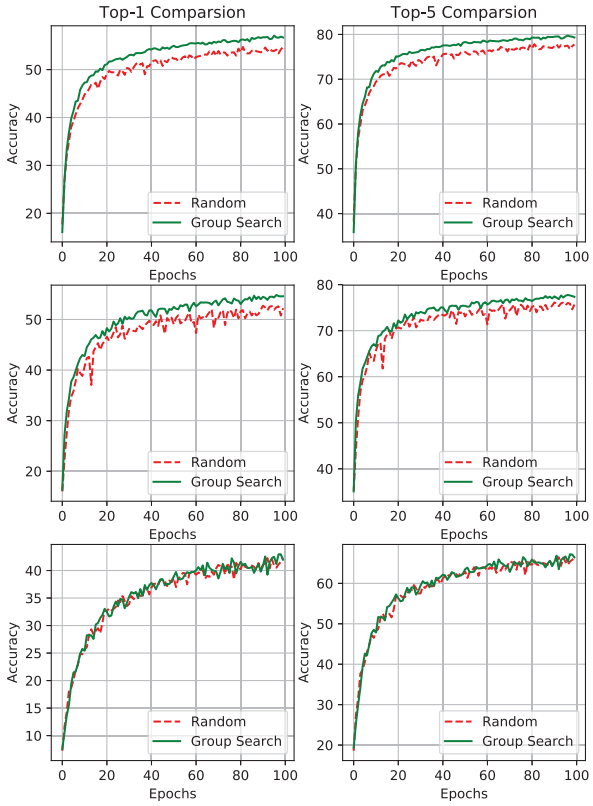

Figure 5. Validation accuracy on ImageNet through epochs of modified modules (M1, M2 and M3, from top to bottom respectively). Random: Groups are naively randomized for layers. Group search: Our proposed search optimal group architecture.

images to train the 1-bit models with random groups and learning rate of 0.001 in 64 epochs. In the search stage, the number of populations, crossovers, mutations are 50, 25, 25 respectively and the searching runs 20 iterations to find optimal group structure. In the end of the pipeline, we train the final models from scratch with batch size 512, learning rate 0.001, number of epochs 256. All training stages use Adam Optimizer [21], momentum 0.9, and update learning rate through linear decay. FLOPs are calculated following the suggestion of [32, 45] for fair comparison. Training is conducted on four RTX 2080 Ti GPUs 24GB and searching is on one GPU.

4.2. Our Search Group vs. Random Group Search and Uniform Group Architectures

To investigate our hypothesis of binarizing convolution via an evolution-based searching in MobileNet's architecture, we compare with random search and uniform group architectures as an ablation study. The experiment is conducted on the three proposed modules in Section 3.2.2. For the Module 2, there are four full-precision 1×1 convolutions. We also apply searched groups for such layers to further reduce the computational cost.

#Groups (M1)	Top-1 (%)	Top-5 (%)	FLOPs
Groups = 1	64.51	85.14	2.13×10^8
Groups = 4	60.89	82.54	1.63×10^8
Groups = 16	58.49	80.66	1.50×10^8
Random Group	59.05	81.22	1.58×10^8
Ours	**60.90**	**82.60**	$\mathbf{1.54} \times 10^8$
#Groups (M2)	Top-1 (%)	Top-5 (%)	FLOPs
Groups = 1	64.51	85.14	2.13×10^8
Groups = 4	59.59	81.67	0.67×10^8
Groups = 16	54.23	77.04	0.30×10^8
Random Group	58.13	80.42	0.75×10^8
Ours	**59.30**	**81.00**	$\mathbf{0.62} \times 10^8$
#Groups (M3)	Top-1 (%)	Top-5 (%)	FLOPs
Groups = 1	57.56	79.85	0.87×10^8
Groups = 4	49.90	73.15	0.37×10^8
Groups = 16	45.29	69.38	0.24×10^8
Random Group	50.07	74.11	0.38×10^8
Ours	**51.06**	**74.18**	$\mathbf{0.33} \times 10^8$

Table 1. Uniform grouping baselines and random group search vs Our group search on Module M1,M2, and M3.

To compare with random group search, we report Top-1 and Top-5 accuracy for each epoch. The training performance comparison is indicated in Figure 5, showing the result of modifications from MobileNets: Module 1, Module 2, and Module 3 (from the top to the bottom in that order).

We run the comparison experiments of randomized and searched groups with 100 epochs, 512 for batch size and observe the Top-1 and Top-5 accuracy in ImageNet validation set for each epoch. With respect to the first two modules, our group search architecture training is more stable and for all epochs, we achieve more accurate results (about 2%) in both Top-1 and Top-5 accuracy.

Our proposed search group achieves better performance when comparing with random groups that require more computational cost. Table 1 reports the number of FLOPs when running with random group and with our proposed group search. Regardless of the fact that random architectures have a larger computational cost, our search group networks are more accurate and efficient.

We also provide a comparison with uniform group (i.e., using the same number groups for all layers of Module M1, M2, and M3) as a ablation study for investigating our hypothesis. We train models with uniform groups of 1 (fully convolution), 4, and 16. The Table 1 presents the results of Top-1, top-5 accuracy, and number of FLOPs (computational cost).

Our reported statistics expresses a trade-off of performance between fully convolution and depth-wise convolution. For example, in M1 and M3 our searched group models outperform comparable uniform group models (g=4 and 16) in accuracy and take less FLOPs.

M	M1	M2	M3
Top-1 (%)	60.9	59.3	51.1
Top-5 (%)	82.6	81.0	74.2
FLOPs	1.54×10^8	0.62×10^8	0.33×10^8
MaxFLOPs	1.55×10^8	0.80×10^8	0.50×10^8
#GPU-hours	30	32	26

Table 2. The efficiency of proposed module M1, M2 and M3 in searched group architecture. The results are conducted on large scale of ImageNet dataset. MaxFlops is the constraint budget.

4.3. The Efficiency of Our BNNs

MobileNet architecture is a compact network working accurately and efficiently based on light-weight module of separable convolution layers. Binarizing such a compact model can give us promising outcomes because it contains less parameters thanks to the tremendous reduction of the computational cost without incurring accuracy loss. However, as mentioned in Section 1 the networks exploiting the separable convolutions including depth-wise scheme cannot convergence when being binarized because of extremely small value range that cannot adequately fit complex data samples like images. On the contrary, groups and fully convolutional layers are easier to make the networks perform well. Albeit achieving high accuracy, fully convolutional layers are not efficient to deploy on mobile devices because of a huge number of parameters. So, group convolutional layers can have potential trade-off between depth-wise and full convolution. In this work, we propose a group search mechanism via evolutionary method to find group structure at each convolutional layers for a binary neural network in the MobileNet architecture.

For showing the effectiveness of our proposed search mechanism, we conduct experiments of modified modules with different computational budget constraints. We firstly train binary models with random groups for each module in 64 epochs. Then, we search for networks satisfying the FLOP budget to derive optimal group structures. Finally, the networks with optimal groups are trained from scratch in 256 epochs. The other settings are mentioned in Section 4.1. Top-1, Top-5 accuracy on ImageNet-1k, FLOPs, budget constraint, and number of GPU-hours of searching are reported in the Table 2.

Compared to the full-precision MobileNet [16], our constructed binary neural networks accelerate approximately $4\times$, $9\times$, and $17\times$ when using Module 1, Module 2, and Module 3 respectively, while incurring small Top1-accuracy loss of 10%, 11.6%, and 19.8%. We also outperform the most related work of MoBiNet [38]. This detail is mentioned in Section 4.4. In addition, our search algorithm only takes $\approx$ 29h on one GPU in average.

In the next section, we compare our method with other state-of-the-art binary neural networks.

4.4. Comparison with State-of-the-art Methods

Binary neural networks make an amazing progress when recently achieving impressive results. However, prior works improve binary models through training process for representation learning while the architecture design should has great influence as well. Our proposed method using evolutionary search based on recent ideas of one-shot neural architecture search aims at exploring the group architecture design for BNNs improvement.

In this section, to evaluate the proposed method we compare our BNNs with several recent works: Binary Connect [8], BNNs [19], ABC-Net [26], DoReFa-Net [53], XNOR-Net [39], etc. The metrics reported are Top-1, Top-5 accuracy on ImageNet, and the number of FLOPs. BiReal-Net and CI-BCNN are two prominent works achieving good results. These networks binarize ResNet [14] with efficient skip connection module. Here, we only consider ResNet 18 layers versus our 13 layers for fair comparison. CI-BCNN [45] is the state-of-the-art binary model (both weights and activations are binarized) as it is able to achieve 59.90% Top-1 accuracy on ImageNet with very small cost of 1.54×10^8 FLOPs. Our binary model using Module 1 outperforms the MoBiNet [38] 6% and the Bi-RealNet-18 [32] 4% Top-1 accuracy with less computational cost. Moreover, it also surpasses CI-BCNN [45] 1% Top-1 accuracy with lower number of FLOPs (ours: 1.54×10^8, CI-BCNN: $> 1.54 \times 10^8$). Also, our Module 2 and Module 3 also transcends the BiReal-Net [32] by requiring a significant lower number of FLOPs.

On the other hand, our method significantly outperforms the other binary neural networks regarding the accuracy and computation metric. The accuracy results are reported in the Table 3. Our proposed binary networks are better than most of the prior works. For computational cost, Table 1 indicates comparisons in terms of number of FLOPs and memory usage.

4.5. Analysis

In this section, we discuss the analysis of results and computational complexity. For ablation study in Section 4.2, the results of group search architecture are more stable and have higher accuracy than naively erratic groups, proving that having heterogeneous group structure at each layer in MobileNet architecture yields good performance. In addition, group convolution is flexible to increase or decrease the number of connections in selective layers. For example when observing the first layers, we realize that the search algorithm tends to assign small number of groups to preserve essential information of the inputs. Meanwhile, the algorithm diminishes insignificant inter-channel connections (i.e., by increasing the number of groups) to enhance the model's compactness and efficiency.

From Table 1 and Table 3, our module 2 and module 3 have higher accuracy than the two prominent works of BiReal-Net [32] and CI-BCNN [45]. Moreover, the modules have a much lower computational cost ($\approx$ 33M FLOPs), approximately $5\times$ speed up factor when comparing with BiReal-Net (163M FLOPs).

Networks	W/A	Top-1	Top-5
Binary Connect [8]	1/32	35.40	61.00
BWN [39]	1/32	56.80	79.40
BNNs [19]	1/1	42.20	67.10
ABC-Net [26]	1/1	42.70	67.60
DoReFa-Net [53]	1/1	43.60	-
SQ-BWN [10]	1/1	45.50	70.60
XNOR-AlexNet [39]	1/1	44.20	69.20
XNOR-ResNet-18 [39]	1/1	51.20	73.20
MoBiNet [38]	1/1	54.40	77.50
Bi-RealNet-18 [32]	1/1	56.40	79.50
CI-BCNN-18 [45]	1/1	56.73	80.12
CI-BCNN-18 (add) [45]	1/1	59.90	84.18
Ours (M3)	1/1	**51.06**	**74.18**
Ours (M2)	1/1	**59.30**	**81.00**
Ours (M1)	1/1	**60.90**	**82.60**

Table 3. The Top-1 and Top-5 accuracy comparison between the state-of-the-art and our method. Our Module 1 (M1) outperforms the state-of-the-art CI-BCNN [45] by 1% Top-1 accuracy.

Networks	FLOPs
XNOR-AlexNet [39]	1.38×10^8
XNOR-ResNet-18 [39]	1.67×10^8
Bi-RealNet-18 [32]	1.63×10^8
CI-BCNN-18 [45]	1.54×10^8
CI-BCNN-18 (add) [45]	$> 1.54 \times 10^8$
MoBiNet [38]	0.52×10^8
Ours (M3)	0.33×10^8
Ours (M2)	0.62×10^8
Ours (M1)	1.54×10^8

Table 4. Computational cost comparison between the state-of-the-arts and our method.

5. Conclusion

Efficient group design for BNNs can yield good outcomes. We introduced a novel algorithm via evolutionary search to explore group structures aiming at optimizing the trade-off when either using depth-wise or fully convolutional layers in MobileNet. Our BNN is efficient as it achieves highly accurate results while saving the computational cost (only single GPUs for searching) in dealing with challenging visual classification tasks.

Acknowledgements. We thanks all anonymous reviewers for constructive and valuable feedback.

References

[1] Yiwen Guo Lin Xu Yurong Chen Aojun Zhou, Anbang Yao. Incremental network quantization: Towards lossless cnns with low-precision weights. In *International Conference on Learning Representations,ICLR2017*, 2017.

[2] Joseph Bethge, Marvin Bornstein, Adrian Loy, Haojin Yang, and Christoph Meinel. Training competitive binary neural networks from scratch. *ArXiv e-prints*, 2018.

[3] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. In *BMVC*, 2019.

[4] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2019.

[5] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tan. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *The International Conference on Computer Vision (ICCV)*, October 2019.

[6] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. Detnas: Backbone search for object detection. In *NeurIPS 2019*, 2019.

[7] Matthieu Courbariaux and Yoshua Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016.

[8] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *NIPS*, pages 3123–3131, 2015.

[9] Jia Deng, Wei Dong, Richard Socher, Li jia Li, Kai Li, and Li Fei-fei. Imagenet: A large-scale hierarchical image database. In *In CVPR*, 2009.

[10] Yinpeng Dong, Renkun Ni, Jianguo Li, Yurong Chen, Hang Su, and Jun Zhu. Stochastic quantization for learning accurate low-bit deep neural networks. *International Journal of Computer Vision*, Mar 2019.

[11] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. *arXiv preprint arXiv:1904.00420*, 2019.

[12] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *CoRR*, abs/1510.00149, 2015.

[13] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 1135–1143, Cambridge, MA, USA, 2015. MIT Press.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[15] Koen G. Helwegen, James Widdicombe, Lukas Geiger, Zechun Liu, Kwang-Ting Cheng, and Roeland Nusselder. Latent weights do not exist: Rethinking binarized neural network optimization. *ArXiv*, abs/1906.02107, 2019.

[16] 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. *CoRR*, abs/1704.04861, 2017.

[17] Gao Huang, Shichen Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Condensenet: An efficient densenet using learned group convolutions. In *The IEEE Conference on CVPR*, June 2018.

[18] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NIPS*, pages 4107–4115, 2016.

[19] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4107–4115. Curran Associates, Inc., 2016.

[20] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡1mb model size. *CoRR*, abs/1602.07360, 2017.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[22] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[24] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[25] Quanquan Li, Shengying Jin, and Junjie Yan. Mimicking very efficient network for object detection. *2017 IEEE Conference on CVPR*, pages 7341–7349, 2017.

[26] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 345–353. Curran Associates, Inc., 2017.

[27] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 19–35, Cham, 2018. Springer International Publishing.

[28] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[29] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient

convolutional networks through network slimming. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2755–2763, 2017.

[30] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Tim Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *The International Conference on Computer Vision (ICCV)*, October 2019.

[31] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019.

[32] 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 *ECCV*, 2018.

[33] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[34] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. Chapter 15 - evolving deep neural networks. In Robert Kozma, Cesare Alippi, Yoonsuck Choe, and Francesco Carlo Morabito, editors, *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293 – 312. Academic Press, 2019.

[35] Wojciech Mula, Nathan Kurz, and Daniel Lemire. Faster population counts using AVX2 instructions. *CoRR*, abs/1611.07612, 2016.

[36] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[37] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4095–4104, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[38] Hai Phan, Dang Huynh, Yihui He, Marios Savvides, and Zhiqiang Shen. Mobinet: A mobile binary network for image classification. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[39] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV (4)*, volume 9908 of *Lecture Notes in Computer Science*, pages 525–542. Springer, 2016.

[40] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *The AAAI Conference on Artificial Intelligence*, 2018.

[41] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V. Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 2902–2911. JMLR.org, 2017.

[42] Anat Caspi Linda Shapiro Sachin Mehta, Mohammad Rastegari and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *ECCV*, 2018.

[43] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381, 2018.

[44] L. Sifre. Rigid-motion scattering for image classification, 2014.

[45] Ziwei Wang, Jiwen Lu, Chenxin Tao, Jie Zhou, and Qi Tian. Learning channel-wise interactions for binary convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[46] L. Xie and A. Yuille. Genetic cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1388–1397, Oct 2017.

[47] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016.

[48] Chris Ying, Aaron Klein, Esteban Real, Eric Christiansen, Kevin Murphy, and Frank Hutter. NAS-Bench-101: Towards Reproducible Neural Architecture Search. *arXiv e-prints*, Feb 2019.

[49] Jiahui Yu and Thomas S. Huang. Universally slimmable networks and improved training techniques. *ArXiv*, abs/1903.05134, 2019.

[50] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas S. Huang. Slimmable neural networks. *CoRR*, abs/1812.08928, 2018.

[51] Jianhao Zhang, Yingwei Pan, Ting Yao, He Zhao, and Tao Mei. dabnn: A super fast inference framework for binary neural networks on arm devices. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2272–2275, 2019.

[52] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018.

[53] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR*, abs/1606.06160, 2016.

[54] Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. Trained ternary quantization. *arXiv preprint arXiv:1612.01064*, 2016.

[55] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.