Instance-Aware Dynamic Neural Network Quantization

Zhenhua Liu[1,2], Yunhe Wang[2], Kai Han[2], Siwei Ma[1,3], Wen Gao[1,3]
[1]National Engineering Research Center of Visual Technology, School of Computer Science, Peking University
[2] Huawei Noah's Ark Lab [3]Peng Cheng Laboratory
{liu-zh,swma,wgao}@pku.edu.cn, {yunhe.wang,kai.han}@huawei.com

Abstract

Quantization is an effective way to reduce the memory and computational costs of deep neural networks in which the full-precision weights and activations are represented using low-bit values. The bit-width for each layer in most of existing quantization methods is static, i.e.*, the same for all samples in the given dataset. However, natural images are of huge diversity with abundant content and using such a universal quantization configuration for all samples is not an optimal strategy. In this paper, we present to conduct the low-bit quantization for each image individually, and develop a dynamic quantization scheme for exploring their optimal bit-widths. To this end, a lightweight bit-controller is established and trained jointly with the given neural network to be quantized. During inference, the quantization configuration for an arbitrary image will be determined by the bit-widths generated by the controller,* e.g.*, an image with simple texture will be allocated with lower bits and computational complexity and vice versa. Experimental results conducted on benchmarks demonstrate the effectiveness of the proposed dynamic quantization method for achieving state-of-art performance in terms of accuracy and computational complexity. The code will be available at https://github.com/huawei-noah/Efficient-Computing and https://gitee.com/mindspore/models/tree/master/research/cv/DynamicQuant.*

1. Introduction

Deep convolutional neural networks (CNNs) have achieved remarkable results in a wide range of intelligent applications including image processing [14, 27], video understanding [5], natural language processing [15] and speech recognition [49]. However, these models have excessive demands on storage and computational resources to achieve satisfactory performance, which prohibits their deployment on mobile and embedded devices. Thus, how to effectively compress and accelerate the CNNs is urgently required for real-world applications.

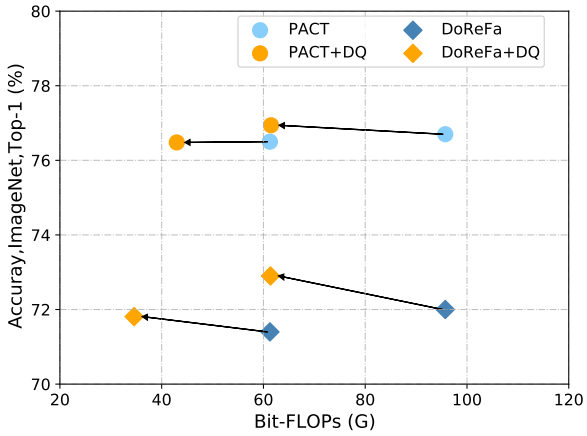

Figure 1. Results of applying DQNet for ResNet50 on the ImageNet dataset. The proposed DQNet can be easily embedded with mainstream quantization frameworks for better performance, such as DoReFa [51] and PACT [8].[1]

Admittedly, there have been extensive explorations on model compression and acceleration methods such as pruning [16, 18, 30, 39], tensor decomposition [24, 38] and knowledge distillation [20, 26], which aims to shrink the original network architectures while remaining their performance. Quantization is another effective approach for reducing the complexity by representing weights and activations in the network using low-bit integer values, which is convenient for applications since it does not change the network architecture and is easily to deploy. For example, Zhou et al. [51] proposed to constrain weights and activations to low-bit values and approximated the sign function with "hard tanh" function [2] in the backward process to avoid the zero-gradient problem. The subsequent methods in [8, 12, 48, 50] utilized flexible scale factors and training strategy to optimize the quantized models for better performance.

[1]The Bit-FLOPs is calculated as the product of the weight bit-width, the activation bit-width and the FLOPs (the multiplication and add operations).

Although the aforementioned approach have made tremendous efforts for enhancing the performance of the low-bit quantized network, the diversity of each instance in the given dataset is usually ignored. In fact, how to effectively handle hard and easy examples is a widespread problem in computer vision, especially in visual recognition task as discussed in [35, 37]. To this end, Wu et al. [44] propose to select a small proportion of layers from the pre-trained network during inference and reduce the overall computational complexity. Cheng et al. [7] point out that a single network architecture is not representative enough for all samples in the given dataset and exploit an instance-level network variation algorithm. These methods are mainly developed based on the fact that the required resources for processing different samples using deep neural networks are often various. For example, a well trained network is very easy to identify an image containing only one dog, but it is hard to recognize an obscured bicycle in the street view. Thus, we are motivated to explore an instance-aware dynamic approach that can provide better trade-off of performance and computational complexity.

In this paper, we propose a novel network quantization scheme, which dynamically allocates bit-widths in quantized neural networks conditioned on each input samples as shown in Fig 2. Specifically, a great number of hidden sub-networks with various bit-width configurations will be derived from the given network architecture. During the inference, an image which is hard to be accurately recognized will be assigned with a larger network and vice versa. For an arbitrary image, a bit-controller is utilized for predicting its optimal bit-width sequence for weights and activations of all layers. The bit-controller is designed with a lightweight architecture so that its additional computational costs can be ignored. The quantized neural network is trained together with the bit-controller in an end-to-end manner for better performance. Since our dynamic quantized network (DQNet) can provide the optimal bit-width configuration according to different input images, the computational cost for processing each sample can be largely reduced. The dynamic computation resource allocation can achieve a better trade-off between computational cost and accuracy than those of conventional static quantization methods. Extensive experiments conducted on CIFAR-10 and ImageNet datasets demonstrate that DQNet can achieve similar or even better classification accuracy than that of static quantization methods while consuming less computation resources.

2. Related works

Here, we will briefly introduce the current works on network quantization including the conventional static quantization and mixed-precision quantization. Besides, the existing dynamic inference methods are summarized and analyzed.

Quantization In order to improve the hardware efficiency, many researchers have proposed to quantize the weights and activations, thus allowing the lower precision computational units in hardware. Courbariaux et al. [9] proposed to present weights and activations with binary values and furthermore they proposed to compute the scaling factor applying to both binary weights and binary inputs in XNOR-Networks [34]. The concept of non-uniform quantization had been suggested by [32] and [1]. Yang et al. [45] formulated quantization function as a linear combination of several sigmoid functions with learnable biases and scales. Esser et al. [13] introduced a means to estimate and scaled the task loss gradient at each weight and activation layer's quantizer step size. These methods utilize the same quantization structure for all the samples and ignore the diversity and complexity of the given dataset.

Mixed-precision Quantization Conventional quantization methods often compress all the layers to the same precision, which may cause significant performance degradation. One possibility to address this problem is to use mixed-precision quantization. Wang et al. [41] proposed a reinforcement learning based method to implement mixed-precision quantization. Dong et al. [10, 11] proposed to decide the quantization bit-widths exploiting second-order (Hessian Matrix) information, which considered the quantization effect of each layer individually. Guo et al. [17] proposed a single path one-shot approach to search the mixed-precision architectures. Differentiable architecture search (DARTS) [28] relaxed the discrete search space into a continuous one, enabling the optimization by gradient descent. Wu et al. [43] and Cai et al. [4] employed DARTS to find the bit allocation for each layer of CNNs. Although these methods quantize each layer with different bit-widths, they do not consider that computation requirement for different samples are various.

Dynamic Inference Recently, there are some dynamic neural networks that focus on designing different architectures under different circumstances. Liu et al. proposed D^2NN [29], which executed a subset neurons of a network given an input. Wang et al. [42] introduced to selectively skip convolutional blocks based on the activations of the previous layer using a gating network. Huang et al. developed MSDNet [21], which trained multiple classifiers with varying resource demands. Yu et al. [47] proposed slimmable neural networks, which learned a single neural network executable at different width. Cai et al. [3] proposed to train a one-for-all network that supports diverse architectural settings by decoupling training and search. Chen

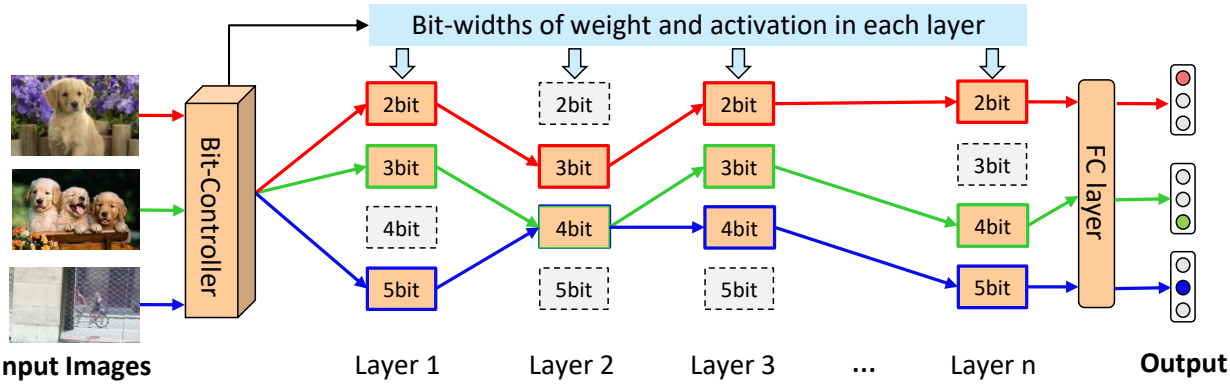

Figure 2. The diagram of the proposed dynamic network quantization approach. The bit-controller first generates the optimal bit-width sequence for each input image and the network will be adjusted to the optimal quantization accordingly.

et al. [6] proposed to aggregate multiple parallel convolution kernels dynamically based upon their attentions. These methods consider either the complexity of the dataset or the limits of different computation resources and further enhance the performance of CNNs. Based on the fact that the required computational complexity for different samples are various, we shall develop an instance-aware quantization algorithm to properly allocate computation resources of quantized neural networks.

3. Approach

In this section, we first revisit the conventional network quantization approach and then describe our dynamic quantization strategy for higher performance in terms of both network accuracy and efficiency.

3.1. Network Quantization

For an arbitrary neural network with n convolutional layers $\{L_1, ..., L_n\}$, there are n sets of weights $\mathbf{W} = \{W_1, ..., W_n\}$ belonging to these layers. For the i-th convolutional layer, its weight is denoted as $W_i \in \mathbb{R}^{c_i \times s_i \times s_i \times o_i}$, where s_i is the kernel size, c_i and o_i are the input and output number of channels in this layer respectively. Given a batch of training samples $\mathbf{X}$ and the corresponding ground truth $\mathbf{Y}$, the error of the network can be defined as $\mathcal{L}_{cls}$, where $\mathcal{L}_{cls}$ could be cross entropy loss or mean squared error, *etc.*. The training of the given deep neural network is conducted by solving an Empirical Risk Minimization problem.

Basically, the training and inference of conventional neural networks utilize floating-point numbers, *i.e.*, both the weights and activations are stored using 32-bit precision. Model quantization methods represent the weights or activations in neural networks with low-bit values so as to reduce the computation and memory costs. To quantize the weights W_i and activations A_i, these floating-point numbers need to be restricted to a finite set of values. The quantization function is usually defined as:

$$Q(z) = \gamma_j, \quad \forall\, z \in (u_j, u_{j+1}], \tag{1}$$

where $(u_j, u_{j+1}]$ denotes a real number interval ($j = 1, ..., 2^b$), b is the quantization bit-width, and z is the input value, *i.e.* a weight or an activation. The quantization function in Eq. (1) maps all the values in the range of $(u_j, u_{j+1}]$ to γ_j. For the choices of these intervals, the widely used strategy is to use an unified quantization function [23,51] in which the above range is equally split, *i.e.*,

$$Q_\Delta(z) = \mathcal{R}(\frac{z}{\Delta}) \cdot \Delta, \tag{2}$$

where the original range (l, r) is divided into 2^b unified intervals, $\Delta = \frac{r-l}{2^b}$ is the interval length and $\mathcal{R}$ is the round function. To make the non-differential quantization process can be optimized end-to-end, the straight-through estimator [2] is usually adopted to approximate the derivative of the quantization function.

3.2. Dynamic Quantization

Although a great number of quantization methods have been explored, the bit-width in conventional quantization method is usually static for all the inputs. In fact, the diversity of natural images in recent datasets is very high and most of existing quantization algorithms do not consider their variousness and intrinsic complexity. To allocate the computation resources precisely, we propose the dynamic quantization to adjust the bit-width for each layer according to the input. Suppose there are K bit-width candidates, *i.e.*, $b^1, b^2, \cdots, b^k$. Dynamic quantization aims to select an optimal bit-width for quantizing weights and activations of each layer, which can be formulated by aggregating multi-

ple bit-widths as follows:

$$Q_{i,j}(z) = \sum_{k=1}^{K} p_{i,j}^k \cdot Q_{\Delta_i^k}(z), \\ s.t.\ p_{i,j}^k \in \{0,1\}, \sum_{k=1}^{K} p_{i,j}^k = 1. \tag{3}$$

where $p_{i,j}^k$ denotes the selection of the k^{th} bit-width for the j^{th} sample in the i^{th} layer. Then the dynamic quantization of j^{th} sample in the i^{th} layer can be formulated as:

$$Y_{i,j} = \widehat{W}_{i,j} * \widehat{X}_{i,j}, \tag{4}$$

$$\widehat{W}_{i,j} = Q_{i,j}(W_i) = \sum_{k=1}^{K} p_{i,j}^k \cdot Q_{\Delta_i^k}(W_i), \tag{5}$$

$$\widehat{X}_{i,j} = Q_{i,j}(X_{i,j}) = \sum_{k=1}^{K} p_{i,j}^k \cdot Q_{\delta_i^k}(X_{i,j}), \tag{6}$$

where Δ_i^k and δ_i^k denote the quantization intervals of weights and activations, respectively and the same bit-width is applied to the weights and activations in one layer. By exploiting Eq. (4)-(6), the dynamic convolutional layer is exactly the aggregation of mixed precision layers for a given input, which can fully exploit the potential of the resulting quantized neural network. Note that the biases are omitted in the formulation for convenience, the quantization interval Δ_i^k of biases are the same with which of weights in practice.

The diagram for using the proposed dynamic quantized network (DQNet) is shown in Figure 2, Given an image, our goal is to provide an optimal trade-off between network performance and computation burden. Thereby, the quantized neural network can still work well under a certain limit of computation. However, Eq. (4)-(6) cannot be directly optimized since the selection indicators $p_{i,j}^k$ for each layer are not fixed, which are variant to the input x. The test dataset cannot be obtained in advance and for the training dataset, the storage requirement for the selection indicators will be tremendous. For example, for ResNet-50 network, the number of possible bit-width configurations is $5^{50} = 8.8 \times 10^{34}$ for each sample when there are five bit-width candidates.

3.3. Bit-Controller for Dynamic Quantization

To address the above challenging problem, we employ a bit-controller to predict the bit-widths of weights and activations of all layers by identifying the complexity of each input sample. In practice, the bit-controller will output a vector consisting of prediction logits, representing the selection probabilities of each bit-width candidate in each layer.

The bit-controller is carefully designed with extremely small architectures to avoid obviously increasing the overall burden on memory and computation of the resulting network. Specifically, the bit-controller is a smaller network

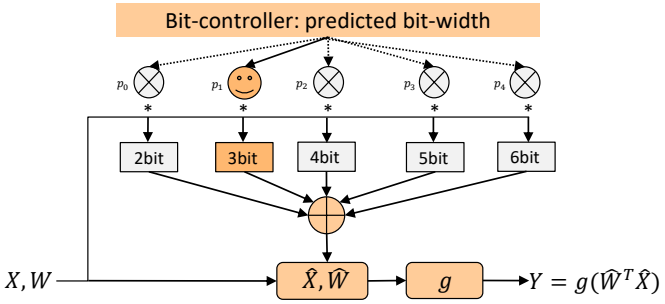

Figure 3. The paradigm of training one layer using dynamic quantization approach. The bit-widths of weights and activations in this layer are predicted by the bit-controller and only one bit-width will be selected for processing the given input.

consisting of the first several layers of the main network followed by a MLP and the MLP consists of only two fully-connected layer in practice. In this way, our DQNet predicts the bit-width of each layer with negligible computation. We examine the impact of employing some layers from the main quantized network for the bit-controller in Sec. 4.4. In addition, the bit-controller is jointly trained with the main quantized network in an end-to-end manner and the bit-controller will generate the prediction logits of bit-widths of all the subsequent layers at once.

Assuming that the output logits of the bit-controller are $h^1, h^2, \cdots, h^k$ for the weights and activations of a certain layer, the bit-width can be selected accordingly, *i.e.*, p^k is determined as

$$p^k = \begin{cases} 1, & \text{if } k = \arg\max_{k'} h^{k'}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

During both training and inference, only one bit-width will be selected for one layer as shown in Figure 3. To provide a differentiable formula for sampling argmax, we utilize the Gumbel-softmax trick during training:

$$p^k = \frac{e^{(h^k+\pi^k)/\tau}}{\sum_j e^{(h^j+\pi^j)/\tau}}, \tag{8}$$

where τ is the temperature parameter that controls how closely the new samples approximate one-hot vectors. π^k is a random noise that follows Gumbel distribution,which can be described as :

$$\pi^k = -log(-log(u^k)),\ u^k \sim U(0,1). \tag{9}$$

During the feed-forward process, after obtaining the bit-width of a certain layer for a specific sample, DQNet will

Algorithm 1 The training procedure of the proposed dynamic quantization scheme.

Input: Input images $\mathbf{X}$ and their labels; the trade-off parameter α.

Output: The dynamic quantized neural network using our algorithm.

Set epochs T and batch size m for training DQNet
Initialize the weights of network and bit-controller randomly and quantize the input images
for $t = 1, \cdots, T$ **do**
  Output the probability of different bit-widths for each layer using the bit-controller.
  Generate the one-hot vectors for selections of different bit-widths using Eq. (7).
  for $i = 1, \cdots, n$ **do**
    for $j = 1, \cdots, m$ **do**
      Quantize $X_{i,j}$ and W_i for j^{th} sample in the i^{th} layer according to the selection of different bit-widths using Eq. (5) and (6).
    end for
    Calculate the output of i-th layer using Eq. (4).
  end for
  Update the entire network using back propagation according to Eq. (11) and utilize Eq. (8) when updating the Gumbel-softmax layer.
end for

quantize the weights and the activations accordingly, *i.e.*,

$$\begin{aligned} \widehat{W}_{i,j} &= Q_{i,j}(W_i) = Q_{\Delta_i^k}(W_i), \\ \widehat{X}_{i,j} &= Q_{i,j}(X_{i,j}) = Q_{\delta_i^k}(X_{i,j}), \\ where \quad k &= \arg\max_{k'} h^{k'}, \end{aligned} \tag{10}$$

where Δ_i^k and δ_i^k are the quantization intervals for weights and activations using the predicted bit-width, respectively. During back-propagation, the quantized network will utilize the gradient computed by Eq. (8). It is worth noting that although there are multiple bit-width candidates, only the biggest bit-width of weight value need to be stored in practice. Since the bottleneck on devices is the inference time, the slightly additive storage of weights can be ignored.

3.4. Optimization

In order to control the computational cost of DQNet flexibly, we add a Bit-FLOPs constriction term in the loss function so the total loss is formulated as:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \alpha \cdot max(\sum_{i=1}^{n} B_i - B_{tar}, 0), \tag{11}$$

where B_i is the Bit-FLOPs of the i-th layer and B_{tar} is the target Bit-FLOPs of the quantized networks. α is the trade-off parameter.

Given a target Bit-FLOPs, the dynamic quantized neural network will capture the inherent variance in the computational requirements of the dataset and allocate optimal bit-widths for different instances and different layers. The training procedure of the proposed dynamic quantization scheme is shown in Algorithm 1.

4. Experimental Results

In this section, we first introduce the training details of DQNet and then present the results of DQNet on CIFAR-10 dataset [25] and ImageNet dataset [36]. In the ablation study, the effect of employing first several layers from the main network for the bit-controller and the influence of the number of layers in bit-controller are presented. Last but not the least, we explore the computation allocation of DQNet among different input instances.

4.1. Training details

Datasets and metrics We evaluate our method on two benchmark classification datasets: CIFAR-10 and ImageNet. The CIFAR-10 dataset consists of $50K$ training images and $10K$ test images, which are labeled for 10 classes. During training and inference, the simple data augmentation like in [19] is utilized, which contains translation and flip. ImageNet dataset contains 1.2 million training images and $50K$ validation images labeled for 1,000 categories. We use translation and flip for the data augmentation and we test on the validation set and report top-1 classification accuracy.

Bit-controller For the bit-controller network, the expression ability will be insufficient if a minor network is adopted. However, the computation burden will be oversized which is against our original intention if a large network is embraced. To achieve a optimal trade-off, we employ the first several layers from the main network for the bit-controller, where a MLP consisting of two fully-connected layers is followed by the output. In this way, the additional computation is negligible for the total network. For example, the additional computation of bit-controller are 1.1% for CIFAR-10 (ResNet-20) and 0.9% for ImageNet (ResNet-50), respectively. Specifically, the output feature maps of *stage one* in ResNet (one convolutional layer and two residual blocks for ResNet20) is utilized as the input of the MLP.

Bit-width candidates As for the bit-width candidates, the values around the target bit-width[2] are selected. For example, the bit-width candidates are {3,4,5,6,7} if the target bit-

[2]For a convenient description, here we replace target Bit-FLOPs with the corresponding bit-width. In practice, target bit-width could be a float number, which is more flexible than conventional quantization methods.

Table 1. Comparison on the performance of proposed DQNet with PACT and DoReFa of ResNet20 on CIFAR-10 dataset. The Top-1 accuracy of the reference full-precision model is 91.6%. Here 'MP' represents for mixed-precision.

	Static Quantization			Dynamic Quantization		
Method	Bit	Bit-FLOPs(G)	Top-1 Accuracy(%)	Bit	Bit-FLOPs(G)	Top-1 Accuracy(%)
DoReFa [51]	3	0.34	89.9	∼3 MP	0.34	90.23
DoReFa [51]	4	0.61	90.5	∼4 MP	0.62	90.79
DoReFa [51]	5	0.95	90.4	∼5 MP	0.96	91.09
Average	–	0.63	90.27	–	0.64	**90.70**
PACT [8]	3	0.34	91.1	∼3 MP	0.36	91.38
PACT [8]	4	0.61	91.3	∼4 MP	0.65	91.60
PACT [8]	5	0.95	91.7	∼5 MP	0.98	92.01
Average	–	0.63	91.37	–	0.65	**91.63**

width is 5. Note that the bit-width candidates are $\{2,3,4\}$ if the target bit-width is 3, since the performance will decline a lot from 1-bit quantization. It is well known that the first convolutional layer and the last fully-connected layer are critical for the recognition and 8-bit quantization is applied for these two layers like the settings in conventional quantization methods.

Implementation details We adopt four quantization methods as the benchmark: DoReFa-Net [51], PACT [8], LQ-Nets [48] and LSQ [12]. We conduct DQNet based on these methods and demonstrate the effectiveness of our approach.

We adopt PyTorch [33] and MindSpore [22] for implementation and utilize stochastic gradient descent (SGD) as the optimizer with momentum of 0.9. For CIFAR-10 dataset, the learning rate starts from 0.1 and is scaled by 0.1 at epoch 60, 120, 180. L2-regularizer with decay of 0.0002 is applied to weight. The mini-batch size of 256 is used and the maximum number of epochs is 200. For ImageNet dataset, the learning rate starts from 0.1 and scaled by 0.1 at epoch 30, 60, 85, 110. L2-regularizer with decay of 10^{-4} is applied to weight. The mini-batch size of 256 is used and the maximum number of epochs is 120. For the hyper-parameter α, it is set to 0.01 for CIFAR-10 and 0.05 for ImageNet, as the classification loss on these two datasets are of different orders of magnitude. All the experiments are conducted on NVIDIA V100 GPUs.

4.2. CIFAR-10

In this section, we present the experimental results of ResNet-20 model on CIFAR-10 dataset. Specifically, the comparison between static quantization and our dynamic quantization are shown in Table 1. For ResNet20, it can be seen that the average top-1 accuracy of DQNet is 0.43% and 0.26% better than DoReFa and PACT under the similar Bit-FLOPs, respectively. Besides, DQNet can save about 40% computation while achieves a even better performance than PACT (91.38% in 0.36G Bit-FLOPs vs. 91.3% in 0.61G Bit-FLOPs). Specifically, the results of DQNet is even better than 32-bit precision neural network whose accuracy is 91.6% and our DQNet can achieve 92.01% accuracy using only 0.98G Bit-FLOPs. More results can be seen in the supplementary materials.

4.3. ImageNet

To further demonstrate the effectiveness of DQNet, we apply it on a much larger dataset–ImageNet. We apply the proposed DQNet on four quantization methods: DoReFa-Net [51], PACT [8], LQ-Nets [48] and LSQ [12]. The experiments are conducted with ResNet-50 model, which consists of a convolutional layer followed by 16 ResNet bottleneck blocks and a final FC layer. The results are shown in Table 2.

DoReFa-Net DoReFa-Net quantizes both weights and activations and also uses low bit-width for parameter gradients. As we can seen, dynamic quantization can achieve 2.16% average top-1 accuracy gain over DoReFa-Net, which is a great progress. Besides, the accuracy of the 3-bit DQNet is even better than the performance of the 5-bit DoReFa-Net.

PACT PACT proposes a parameterized clipping activation function and automatically optimizes the quantization scales during model training. PACT is a much better benchmark and we also conduct DQNet based on PACT. The results show that DQNet still can achieve remarkable profit, which is 0.45% average Top-1 accuracy. It is worth noting that dynamic quantization can save 35% computation while achieving a better top-1 accuracy (76.94% in 61.49G Bit-FLOPs vs. 76.7% in 95.73G Bit-FLOPs).

LQ-Nets LQ-Nets is a classical non-uniform quantization method, where DoReFa-Net and PACT are both uniform quantization methods. It apply learnable quantizers which can be jointly trained with the network parameters. The

Table 2. Comparison on the performance of proposed DQNet with DoReFa-Net, PACT, LQ-Nets and LSQ of ResNet50 on ImageNet dataset. The Top-1 accuracy of the reference full-precision model is 76.96%. Here 'MP' represents for mixed-precision.

	Static Quantization			Dynamic Quantization		
Method	Bit	Bit-FLOPs(G)	Top-1 Accuracy(%)	Bit	Bit-FLOPs(G)	Top-1 Accuracy(%)
DoReFa [51]	3	34.46	69.9	∼3 MP	34.68	71.81
DoReFa [51]	4	61.27	71.4	∼4 MP	61.38	72.90
DoReFa [51]	5	95.73	71.4	∼5 MP	95.86	74.48
Average	–	63.82	70.9	–	63.97	**73.06**
PACT [8]	3	34.46	75.3	∼3 MP	34.70	75.81
PACT [8]	4	61.27	76.5	∼4 MP	61.49	76.94
PACT [8]	5	95.73	76.7	∼5 MP	96.27	77.12
Average	–	63.82	76.17	–	64.15	**76.62**
LQ-Nets [48]	3	34.46	74.2	∼3 MP	34.75	74.89
LQ-Nets [48]	4	61.27	74.89	∼4 MP	61.39	75.72
Average	–	47.86	74.54	–	48.07	**75.31**
LSQ [12]	3	34.46	75.8	∼3 MP	34.69	76.16
LSQ [12]	4	61.27	76.7	∼4 MP	61.35	2 3 77.02
Average	–	47.86	76.17	–	48.02	**76.59**

Table 3. Comparison on the performance of proposed DQNet with mixed-precision methods HAQ and HAWQ using ResNet-50 on ImageNet dataset.

Method	model	W-Bits	A-Bits	Bit-FLOPs(G)	Top-1 (%)
HAQ [41]	ResNet-50	∼3 MP	∼3 MP	∼34.46	75.30
HAWQ [11]	ResNet-50	∼2 MP	∼4 MP	∼31.67	75.48
Ours	ResNet-50	∼3 MP	∼3 MP	34.59	**75.81**
HAQ [41]	ResNet-50	∼4 MP	∼4 MP	∼61.27	76.14
Ours	ResNet-50	∼4 MP	∼4 MP	61.49	**76.94**
HAQ [41]	MobileNet-V2	∼3 MP	∼3 MP	∼2.72	70.90
Ours	MobileNet-V2	∼3 MP	∼3 MP	2.81	**71.56**
HAQ [41]	MobileNet-V2	∼4 MP	∼4 MP	∼4.85	71.47
MPDNN [40]	MobileNet-V2	∼4 MP	∼4 MP	–	69.74
AutoQ [31]	MobileNet-V2	4.14	3.67	–	70.8
FracBits [46]	MobileNet-V2	∼4 MP	∼4 MP	5.33	71.3
Ours	MobileNet-V2	∼4 MP	∼4 MP	4.94	**72.05**

experimental results prove the universality of our method. As a result, DQNet promotes the accuracy of LQ-Nets by 0.69% and 0.83% under 3-bit and 4-bit quantization respectively.

LSQ LSQ introduces a novel means to estimate and scale the task loss gradient at each weight and activation layer's quantizer step size, such that it can be learned in conjunction with other network parameters. We can seen that the LSQ achieves the best performance among these methods. However, it can still benefit from the dynamic quantization. It is worth noting that the top-1 accuracy of DQNet based on LSQ is even larger than the full-precision model. We think the reason is that dynamic quantization provides a better regularization for the neural networks.

Comprison with mixed-precision methods We also compare our DQNet with the mixed-precision methods. The results are shown in Table 3. Compared to HAWQ [11], DQNet achieves a better performance under the similar computation cost. Note that the bit-width of weights and activations in HAWQ are different, which is difficult to achieve the theoretical acceleration in practice. Compared to HAQ [41], DQNet performs better under about 35G and 61G Bit-FLOPs. With a more compact model MobileNet-V2, our DQNet outperforms HAQ by 0.66% and 0.58% under about 3-bit and 4-bit quantization, respectively.

4.4. Ablation study

Employing the first several layers of the main network as the front part of bit-controller can obviously reduce the ad-

Figure 4. Qualitative results of DQNet on different samples using various Bit-FLOPs. The context of input images are gets complicated from top to bottom and the consuming computation resources becomes larger.

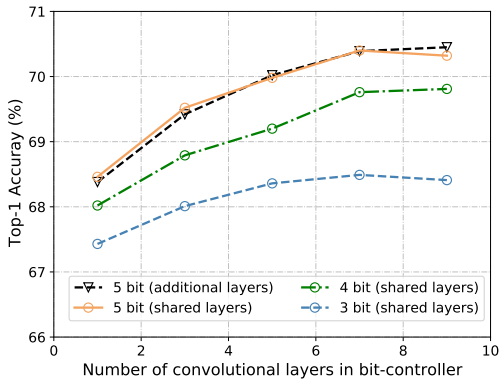

Figure 5. Comparison between using additional convolutional layers and employing convolutional layers from the main network in the bit-controller.

ditional computation cost. We compare the results with using separate convolutional layers as the bit-controller. The experiments are conducted with ResNet18 on ImageNet dataset. As shown in Figure 5, employing the first several layers of the main network achieves comparable results compared to using separate convolutional layers which introduces additional computation costs.

We also test how the number of shared layers between the main network and the bit-controller affect the performance of DQNet. The experiments are also carried out with ResNet18 on ImageNet dataset. As we see in Figure 5, the results of utilizing only one convolutional layer is terrible, which are even worse than static quantization. The reason is that the expression ability of bit-controller is not powerful enough. Along with the increase of the number of layers in bit-controller, the performance of DQNet becomes better. However, the shared layers cannot be too large to have a negative effect on the variety and regularization of the main network.

4.5. Qualitative Results

Finally, the qualitative results based on our learned dynamic quantization strategies are provided. Figure 4 illustrates some samples from ImageNet dataset. The top row shows the images that are correctly classified with less Bit-FLOPs, while the sample in the bottom row utilize more Bit-FLOPs. It is shown that samples using fewer Bit-FLOPs are indeed easier to identify since they contain single frontal-view objects positioned in the center, while occlusion or cluttered background occur in samples that require more computation.

5. Conclusion

In this paper, we develop a novel dynamic quantization scheme in which the bit-widths of weights and activations in each layer are variant to the input instance. To this end, we propose to use a lightweight bit-controller jointly trained with the entire network. The bit-controller will predict the optimal bit-widths of all layers for maximally exploiting the redundancy of the given network architecture for each input image. Specifically, images with lower recognition complexity will be assigned with a portable network and heavy networks will be employed on others for preserving the recognition accuracy. Experimental results show that the proposed DQNet can be easily embedded into mainstream quantization frameworks for better results in terms of both network accuracy and computation costs. We think that the benefits of the proposed method will be heuristic for the developing of specific hardware.

References

[1] Chaim Baskin, Eli Schwartz, Evgenii Zheltonozhskii, Natan Liss, Raja Giryes, Alex M Bronstein, and Avi Mendelson. Uniq: Uniform noise injection for non-uniform quantization of neural networks. *arXiv preprint arXiv:1804.10969*, 2018. 2

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 1, 3

[3] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019. 2

[4] Zhaowei Cai and Nuno Vasconcelos. Rethinking differentiable search for mixed-precision neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2349–2358, 2020. 2

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 1

[6] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11030–11039, 2020. 3

[7] An-Chieh Cheng, Chieh Hubert Lin, Da-Cheng Juan, Wei Wei, and Min Sun. Instanas: Instance-aware neural architecture search. In *AAAI*, pages 3577–3584, 2020. 2

[8] Jungwook Choi. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. 1, 6, 7

[9] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016. 2

[10] Zhen Dong, Zhewei Yao, Yaohui Cai, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. *arXiv preprint arXiv:1911.03852*, 2019. 2

[11] Zhen Dong, Zhewei Yao, Amir Gholami, Michael Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 293–302, 2019. 2, 7

[12] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019. 1, 6, 7

[13] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019. 2

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 1

[15] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309, 2017. 1

[16] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *NIPS'16 Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 1387–1395, 2016. 1

[17] 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. 2

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

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016. 5

[20] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[21] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q Weinberger. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*, 2017. 2

[22] Huawei. Mindspore. http://www.mindspore.cn/, 2020. 6

[23] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18(1):6869–6898, 2017. 3

[24] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. In *ICLR 2016 : International Conference on Learning Representations 2016*, 2016. 1

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[26] Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong. Learning small-size dnn with output-distribution-based criteria. In *INTERSPEECH*, pages 1910–1914, 2014. 1

[27] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 1

[28] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 2

[29] Lanlan Liu and Jia Deng. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. *arXiv preprint arXiv:1701.00299*, 2017. 2

[30] Zhenhua Liu, Jizheng Xu, Xiulian Peng, and Ruiqin Xiong. Frequency-domain dynamic pruning for convolutional neural networks. In *NIPS 2018: The 32nd Annual Conference on Neural Information Processing Systems*, pages 1043–1053, 2018. 1

[31] Qian Lou, Feng Guo, Lantao Liu, Minje Kim, and Lei Jiang. Autoq: Automated kernel-wise neural network quantization. *arXiv preprint arXiv:1902.05690*, 2019. 7

[32] Eunhyeok Park, Sungjoo Yoo, and Peter Vajda. Value-aware quantization for training and inference of neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 580–595, 2018. 2

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[34] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016. 2

[35] Henry A Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38, 1998. 2

[36] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 5

[37] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016. 2

[38] Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, and Weinan E. Convolutional neural networks with low-rank regularization. In *ICLR 2016 : International Conference on Learning Representations 2016*, 2016. 1

[39] Yehui Tang, Yunhe Wang, Yixing Xu, Yiping Deng, Chao Xu, Dacheng Tao, and Chang Xu. Manifold regularized dynamic network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5018–5028, 2021. 1

[40] Stefan Uhlich, Lukas Mauch, Fabien Cardinaux, Kazuki Yoshiyama, Javier Alonso Garcia, Stephen Tiedemann, Thomas Kemp, and Akira Nakamura. Mixed precision dnns: All you need is a good parametrization. *arXiv preprint arXiv:1905.11452*, 2019. 7

[41] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8612–8620, 2019. 2, 7

[42] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018. 2

[43] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*, 2018. 2

[44] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8817–8826, 2018. 2

[45] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7308–7316, 2019. 2

[46] Linjie Yang and Qing Jin. Fracbits: Mixed precision quantization via fractional bit-widths. *arXiv preprint arXiv:2007.02017*, 1:2, 2020. 7

[47] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. *arXiv preprint arXiv:1812.08928*, 2018. 2

[48] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 365–382, 2018. 1, 6, 7

[49] Zixing Zhang, Jürgen Geiger, Jouni Pohjalainen, Amr El-Desoky Mousa, Wenyu Jin, and Björn Schuller. Deep learning for environmentally robust speech recognition: An overview of recent developments. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 9(5):1–28, 2018. 1

[50] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017. 1

[51] 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. *arXiv preprint arXiv:1606.06160*, 2016. 1, 3, 6, 7