# Once Quantization-Aware Training: High Performance Extremely Low-bit Architecture Search

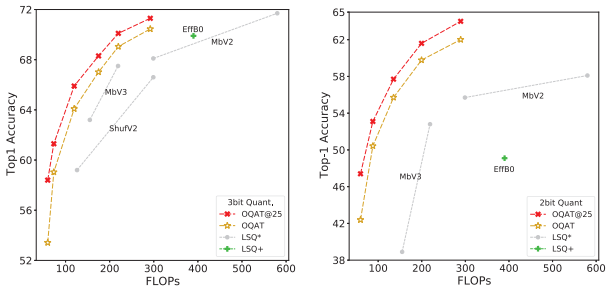
Mingzhu Shen[1] Feng Liang[1] Ruihao Gong[1] Yuhang Li[1] Chuming Li[1]
Chen Lin[2] Fengwei Yu[1] Junjie Yan[1] Wanli Ouyang[3]
[1]Sensetime Research [2]University of Oxford [3]The University of Sydney
shenmingzhu@sensetime.com, liangfeng@sensetime.com, wanli.ouyang@sydney.edu.au

## Abstract

*Quantization Neural Networks (QNN) have attracted a lot of attention due to their high efficiency. To enhance the quantization accuracy, prior works mainly focus on designing advanced quantization algorithms but still fail to achieve satisfactory results under the extremely low-bit case. In this work, we take an architecture perspective to investigate the potential of high-performance QNN. Therefore, we propose to combine Network Architecture Search methods with quantization to enjoy the merits of the two sides. However, a naive combination inevitably faces unacceptable time consumption or unstable training problem. To alleviate these problems, we first propose the joint training of architecture and quantization with a shared step size to acquire a large number of quantized models. Then a bit-inheritance scheme is introduced to transfer the quantized models to the lower bit, which further reduces the time cost and meanwhile improves the quantization accuracy. Equipped with this overall framework, dubbed as Once Quantization-Aware Training (OQAT), our searched model family, OQATNets, achieves a new state-of-the-art compared with various architectures under different bit-widths. In particular, OQAT-2bit-M achieves 61.6% ImageNet Top-1 accuracy, outperforming 2-bit counterpart MobileNetV3 by a large margin of 9% with 10% less computation cost. A series of quantization-friendly architectures are identified easily and extensive analysis can be made to summarize the interaction between quantization and neural architectures. Codes and models are released at https://github.com/LaVieEnRoseSMZ/OQA*

## 1. Introduction

Quantization Neural Networks (QNN) is a promising research direction to deploy deep neural networks on edge devices. Extensive efforts have been devoted to improving quantization performance with Quantization-Aware Training [4, 41, 10, 20, 9, 2, 15] or Post-Training Quantization [28, 21]. Recent studies [27, 26] investigate the quantization impact on architecture and thus take the architecture perspective to pursue high-performance quantized models with expert efforts and manual design. Compared with the laborious manual trials, the combination of Network Architecture Search (NAS) and quantization seems to be a more natural solution.




(a) 3-bit. (b) 2-bit.

Figure 1. Comparison with the state-of-the-art extremely low-bit neural networks on the ImageNet dataset. Our OQATNets achieve a new state-of-the-art under various bit-widths.

The existing combination of NAS and quantization methods could either be classified as NAS-then-Quantize or Quantization-aware NAS as shown in Figure 2. NAS-then-Quantize (Figure 2(a)) usually results in sub-optimal performance because the ranking order of full precision networks is not identical to that of quantized networks. Thus, this traditional routine may fail to get a good quantized model. Directly searching with quantized models' performance (Figure 2(b)) seems to be an alternative. However, due to the instability brought by quantization-aware training, simply combining quantization and NAS results in inferior performance and sub-optimal quantized models as explained in [3]. Moreover, when quantized into 2-bit, the traditional training process is highly unstable and introduces very large accuracy degradation.

Furthermore, all the existing methods [34, 31, 3, 11, 35]

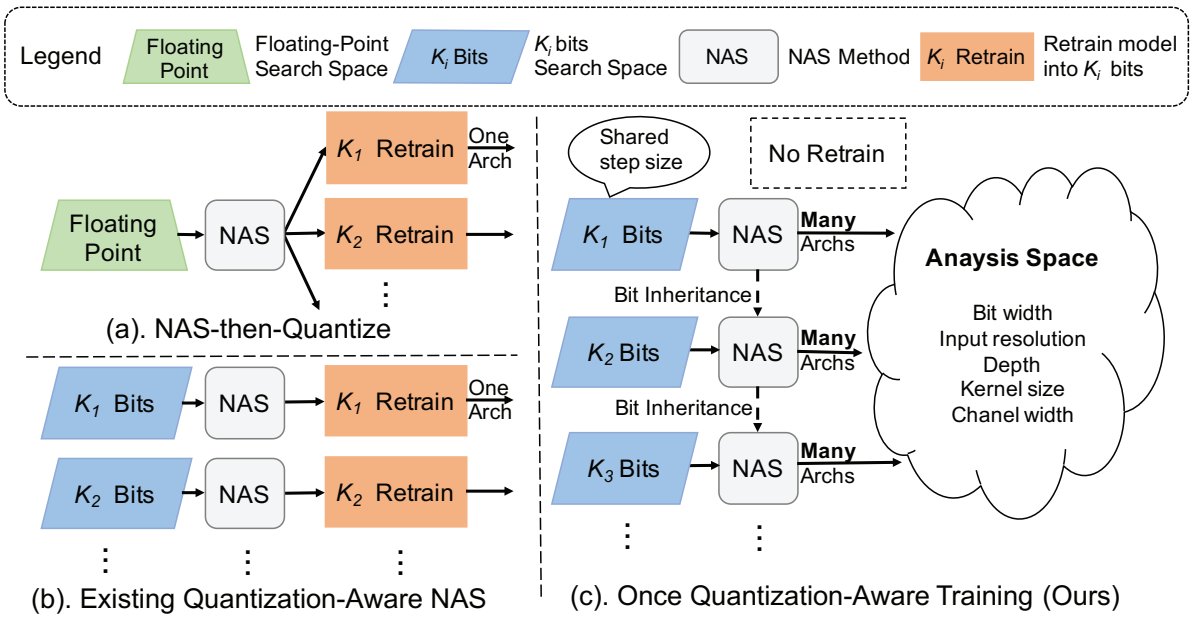



Figure 2. The overall framework of existing works on combining quantization and NAS methods. (a) NAS-then-Quanztize denotes directly converting the best searched floating-point architecture to quantization. (b) Existing Quantization-aware NAS first adopts a quantization-aware search algorithm to find a single architecture, then retrain the quantized weights and activation. (c) Our OQAT can search for many quantized compact models under various bit widths and deploy their quantized weights directly.

adopt a two-stage search-retrain scheme. Specifically, they first search for one architecture under the full-precision or low-bit setting and then retrain the model given the specific bit widths and architecture settings. This two-stage procedure undesirably increases the search and retrain cost if we have multiple deployment constraints and hardware bit widths.

To alleviate the aforementioned problems, we present Once Quantization-aware Training (OQAT), a framework that 1) trains quantized supernet with shared step size and deploys their quantized weights immediately without retraining, 2) progressively produces a series of quantized models under different bit-widths (*e.g.* 4/3/2 bit). Our approach leverages the recent NAS approaches which do not require retraining [37, 5, 38, 6] and combines it with quantization by shared step size. The search space for compact QNN includes kernel size, depth, width, and resolution. To provide a better initialization and transfer the knowledge of the higher bit-width to the lower bit-width, we propose a bit inheritance mechanism, which reduces the bit-width progressively to enable efficient searching for QNN under different quantization bit-widths. Benefiting from the non-retrain property and large search space under different bit widths, we conduct an extensive investigation on the interaction between neural architecture and model quantization, to shed light on the design principles of the quantized network.

Extensive experiments prove the effectiveness of our approach as shown in Figure 1. Our searched quantized model family, OQATNets, achieves state-of-the-art results on the ImageNet dataset under 4/3/2 bit. In particular, our OQAT-2bit-M far exceeds the accuracy of 2-bit MobileNetV3@LSQ [9] by a large 9% margin while using 10% less computation budget. Compared with the quantization-aware NAS method APQ [35], our OQAT-MBV2-4bit-L uses 43.7% less computation cost while maintaining the same accuracy as APQ-B.

To summarize, the contributions of our paper are threefold:

- Our OQAT is a quantization-aware NAS framework with the support of share step size to search for quantized compact models and deploy their quantized weights without retraining.
- We present the bit inheritance mechanism to reduce the bit-width progressively so that the higher bit-width models can guide the search and training of lower bit-width models.
- We provide insights into quantization-friendly architecture design. A comprehensive and systematic study reveals the favored quantization-friendly pattern under different bit widths.

## 2. Related Work

**High-Performance QNN.** QNN has been widely used for efficiency in deployment. Extensive efforts have been devoted to improving the quantization performance of one

Table 1. ImageNet performance under FP,4,3,2 bit with the same settings of quantization-aware training. Model-A and Model-B are two MobileNetV3-like models.

| Models | FLOPs | FP | 4-bit | 3-bit | 2-bit |
|---|---|---|---|---|---|
| ResNet18 | 1813M | 71.0 | 71.0 | 70.2 | 67.6 |
| MobileNetV2 | 299M | 72.0 | 71.3 | 68.2 | 55.7 |
| Model-A | 283M | 75.0 | 72.3 | 68.9 | 54.6 |
| Model-B | 291M | 75.5 | 72.3 | 67.8 | 49.5 |

given architecture. [9] proposes to learn step size in the quantization-aware training (QAT) [41, 7, 2, 22], while some other research achieve this goal by reduce rounding error [28, 21] with small computation cost in the post-training quantization (PTQ). These methods have achieved great progress in high bit widths like 8 with PTQ or under extremely low bit widths for heavy networks like ResNet18 with QAT. The quantization of efficient models [13, 23] still causes large accuracy degradation. Recent works [27, 26] also find a strong correlation between architecture and quantization and intend to push the quantization performance by manual design.

**Quantization-aware Network Architecture Search** Inspired by recent progress of NAS studies like [25, 11, 24, 19, 18, 40, 38, 5], recent studies combine network quantization and NAS to automatically search for layer bit-width with given architecture or search for operations with given bit-width. HAQ [34] focuses on searching for different bit widths for different layers in a given network structure and shows that some layers, which can be quantized to low bits, are more robust for quantization than others. AutoBNN [31] utilizes the genetic algorithm to search for network channels and BMobi [29] searches for the group number of different convolution layers under a certain 1-bit. SPOS [11] trains a quantized one-shot supernet to search for bit-width and network channels for heavy ResNet [12]. BATS [3] devises a binary search space and incorporates it within the DARTS framework [25]. APQ [35] trains a floating-point supernet and samples thousands of subnet for quantization to enable the transfer learning from the floating-point predictor to quantization predictor. Moreover, inherit a two-stage search-retrain scheme: once the best-quantized architectures have been identified, they need to be retrained for deployment. This procedure significantly increases the computational cost if we have different deployment constraints and hardware bit widths.

# 3. Preliminary

## 3.1. Quantization

Quantization maps the floating-point values into fix-point ones. In this paper, we choose uniform quantization since it is widely used in the practical deployment. Given a pre-defined $k$ bit, the weights and activation are quantized to corresponding signed $[-2^{k-1}, 2^{k-1}-1]$ and unsigned $[0, 2^k-1]$ integer range, respectively. The quantization function $Q$ can be formulated as:

$$\mathbf{v}^q = Q(\mathbf{v}, s) = \lfloor \text{clip}(\frac{\mathbf{v}}{s}, Q_{min}, Q_{max}) \rceil \times s, \quad (1)$$

where $\mathbf{v}$ represents the floating-point number and $\mathbf{v}^q$ is the quantized counterpart. $s$ is the step size for quantization mapping and $[Q_{min}, Q_{max}]$ represents the integer range.

To acquire the quantization accuracy, Post-Training Quantization and Quantization-Aware Training are two potential approaches. However, PTQ usually fails to achieve acceptable performance under the extremely low-bit setting [28, 21], which prevents us from revealing the internal quantization friendliness of a neural network. Thus we utilize QAT and adopt the Learned Step Size Quantization [9] to ensure the reported accuracy represents the highest performance of a quantized network. LSQ directly optimizes the step size using the loss:

$$\frac{\partial \mathbf{v}^q}{\partial \mathbf{v}} \approx \mathbb{I}(\mathbf{v}, Q_{min} \times s, Q_{max} \times s) \quad (2)$$

$$\frac{\partial \mathbf{v}^q}{\partial s} \approx -\mathbb{I}(\frac{\mathbf{v}}{s}, Q_{min}, Q_{max}) + \lfloor \frac{\mathbf{v}}{s} \rceil, \quad (3)$$

where $\mathbb{I}(\mathbf{v}, Q_{min} \times s, Q_{max} \times s)$ means the gradient of $\mathbf{v}$ in the range of $(Q_{min} \times s, Q_{max} \times s)$ is approximated by 1, otherwise 0.

## 3.2. Architecture Impact on Quantization

As broadly investigated in the previous literature, architecture plays a crucial role in quantization performance. WRPN [27] improves the quantized accuracy by increasing the width of a neural network by manual design. ReActNet [26] introduces a new activation module and brings new possibility for BNNs.

To make a deeper analysis into the interaction between architecture and quantization, we further conduct more validations. In Table 1, we compare two widely used models ResNet18 and MobileNetV2. Although MobileNetV2 surpasses the accuracy of ResNet18 in full-precision, the accuracy of 2-bit MobileNetV2 reduces significantly. It is common sense that compact models with depthwise convolution like MobileNetV2 tend to be more sensitive to quantization than heavy networks like ResNet18. We further present two MobileNetV3-like models with different depth, width, and kernel size settings. With a similar computation budget, Model-B surpasses Model-A with 0.5% accuracy gain in floating-point settings, while the accuracy of Model-B is 5% lower than Model-A in 2-bit. It indicates that different architecture results in different quantization performances even when they share similar FP accuracy.

## 4. Methodology

In this section, we present Once Quantization-Aware Training (OQAT), a framework that jointly trains quantization and search architectures within a huge search space.

### 4.1. Once Quantization-Aware Training

We first review two prevalent NAS + Quantization routes: (1) *NAS-then-Quantize* searches full precision model performance and then quantizes the optimal FP model as shown in Figure 2(a). This type of method may result in sub-optimal architecture since the quantization-friendliness is not considered during the search process. (2) *Quantization-aware NAS* proposes to incorporate quantization-aware performance in the NAS process as shown in Figure 2(b). However, the simple combination of NAS and quantization results in unstable training or inferior performance as explained in BATS [3]. To stabilize the training process, [3] proposes to only quantize activation which does not search the global quantized architectures. Moreover, both NAS-then-Quantize and Quantization-aware NAS require a two-stage search-retrain scheme, which is unaffordable if we have multiple deployment requirements.

To this end, we propose Once Quantization-Aware Training (OQAT), a framework that jointly trains quantization and searches architecture with a large search space inspired by recent non-retrain NAS methods [39, 37, 5, 38]. Specifically, a quantized supernet with the largest possible depth (number of blocks), width (number of channels), kernel size, and input resolution is trained. Then a subnet is obtained from parts of the supernet with depth, width, and kernel size smaller than the supernet. The subnet uses the well-trained parameters of the supernet with simple BatchNorm calibration [37] for direct deployment without further retraining.

The overall procedure of OQAT is illustrated as follows: Step 1, quantized supernet training (Section 4.2): train a $k$-bit supernet by learning the weight parameters and step size simultaneously. Step 2: given a constraint on computational complexity, search for the architecture with the highest quantization performance on the validation dataset. If $k = n$, the whole process is finished. Step 3, bit inheritance (Section 4.3): Use the weight and quantization parameters of the $k$ bit supernet to initialize the weight and quantization parameters of the $k-1$ bit supernet. Step 4: $k \leftarrow k-1$ and Go to step 1.

Joint training of quantization and architecture does not come as free lunch as it might appear, but is more subtle and involves new designed techniques. Unlike the floating-point supernet training [5, 38], the weights and activation values are quantized with Eq. 1 for the quantized supernet training. As explained in LSQ [9] and recent quantization robustness papers [32, 1], learned step size is sensitive to training, and optimal step size is critical for final quantization accuracy. The design of the learned step size should be carefully handled. Under ultra-low bit-width, the traditional QAT becomes highly unstable and introduces large accuracy degradation. How to produce low-bit quantized supernet with high performance efficiently is also crucial.

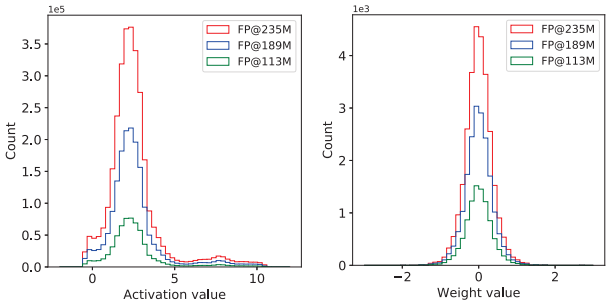



Figure 3. The rationality of shared step size. We randomly pick three subnets from the floating-point (FP) supernet. The subnet is denoted as FP@FLOPs. We depict the activation and weights of the second pointwise convolution layer in the third stage. We find that although the subnets have quite different FLOPs, their activation and weights share almost the same range (x-axis).

### 4.2. Shared Step Size

Quantization step size is essential for quantization error, as it balances the rounding and clipping error. There are 3 potential approaches to handle learned step size:

1. Assign every subnet with unique step sizes for both weights and activation. However, it is impossible to store the huge amount of step sizes for $10^{20}$ subnets.

2. Apply switchable step size for every possible candidate choice in one layer inspired by Switchable BN [39] and Switchable Clipping Level [16]. For example, if the kernel size of one layer has three search settings with $\{3, 5, 7\}$, the number of step sizes is also set to three.

3. Equip every layer with only one step size for weights and one step size for activation.

We first plot the distribution of weights and activation of floating-point supernet and find that the range and distribution are similar among different subnets as shown in Figure 3. It indicates that shared step size is perhaps enough to share across subnets. We also experiment with the latter two methods to support quantized supernet training with elastic kernel size and they indeed result in similar results (see Appendix). And for the simplicity of efficient training, we propose to share the step size across one layer.

Therefore, the forward pass of the quantized supernet training is defined as follows. Given a floating-point convolution layer for candidate architecture $arch_i$, the floating-

Table 2. The accuracy of the biggest model with adaptation to lower bit-width. 4-bit denoted the 4-bit models after individual QAT training, OQAT-4bit denotes quantized supernet at 4-bit. Calib denotes BatchNorm and step size calibration with little computation overhead. 4/3 represents the bit width of weights and activation is 4 and 3 respectively.

| Models | Methods | 4/4 | 4/3 | 3/4 | 3/3 |
|---|---|---|---|---|---|
| 4-bit | Calib | 75.0% | 72.9% | 69.2% | 65.3% |
| OQAT-4bit | Calib | 75.5% | 73.8% | 72.1% | 69.1% |

Table 3. The accuracy of the biggest model in (QAT) and progressive bit inheritance. Start and End denote the accuracy at the first epoch and the end of the training.

| Methods | 4/4 | 3/3 | 2/2 |
|---|---|---|---|
| QAT@Start | 48.1% | 23.2% | 0.8% |
| QAT@End | 75.1% | 72.1% | 56% |
| Bit-Inheritance@Start | - | 71.7% | 49.3% |
| Bit-Inheritance@End | - | 72.7% | 64.5% |

point weights $\mathbf{w}$ and activation $\mathbf{a}$ and the learned step size $s_{\mathbf{a}}$ and $s_{\mathbf{w}}$ are shared through all candidate architectures in this layer. In the forward pass, the weights $\mathbf{w}_i$ and activation $\mathbf{a}_i$ are cropped respectively. The learned quantization function defined in Eq. 1 is used to quantize weights and activation into $\mathbf{w}_i^q$ and $\mathbf{a}_i^q$. The forward pass for a quantized convolution layer is then given by $\mathbf{y} = \mathbf{w}_i^q * \mathbf{a}_i^q$ where $*$ is the convolution.

### 4.3. Bit Inheritance

When the bit-width is lower than 3, the traditional quantization-aware training (QAT) [17, 2] process is highly unstable and introduces large accuracy degradation for the 2-bit model. Besides, despite the higher efficiency of OQAT compared with existing methods, the time cost is still unacceptable when we need to train a supernet for each bit-width.

To compensate for the disadvantages, we devise a bit inheritance procedure, which serves as a good initialization for low-bit models and improves the final QAT accuracy efficiently. We propose that in the initialization for $k-1$ bit supernet, the weight parameters and step size are inherited and calibrated from $k$ bit supernet. Specifically, the step size of $k-1$ bit-width is calibrated with double the original step size. With this simple principle, we can promise a bounded error when transferring the bit-width between supernets. Given a convolution layer in the $k$ bit quantized model, we denote $s_k$ as the step size of this layer, and $s_{k-1} = 2s_k$ as the doubled step for the $k-1$ bit model. We use $\mathbf{w}$ and $N_{\mathbf{w}}$ to denote the weights of this layer which is inherited from $k$ to $k-1$. Next, we show that the $L_1$ distance of $Q(\mathbf{w}, s_k)$ and $Q(\mathbf{w}, s_{k-1})$ is bounded by $N_{\mathbf{w}} \cdot s_k$. It means the initialized $Q(\mathbf{w}, s_{k-1})$ has a bounded distance with the well-trained $Q(\mathbf{w}, s_k)$. For each $w_i$, we have:

$$||Q(\mathbf{w}, s_k) - Q(\mathbf{w}, s_{k-1})||_1 \leq N_w \cdot |s_k|. \quad (4)$$

The detailed proof can be seen in the Appendix. The theorem indicates that our bit inheritance scheme can provide a better initialization for the later finetuning.

**Effectiveness of Bit Inheritance** To validate the effectiveness of bit inheritance, we first directly use it to convert the 4-bit quantized supernet to 3 bit with simple BN and step size calibration. We surprisingly find that the accuracy has already outperformed the result with a complete QAT for the same network (see Table 2). Besides the bounded error brought by the bit inheritance itself, we conjecture that our shared step size and shared parameter training also contribute to the robust transfer. Further, we finetune the inherited 3-bit supernet for just one epoch, the accuracy can effortlessly improve by 2.6%, which also shows that the inherited supernet has already been around the global minimum. Even for the lower bit-width such as 2-bit, it can achieve accuracy close to 3-bit with just a few more epochs' finetuning after inheritance (see Table 3). All the evidence proves that our bit inheritance technique is efficient and effective for producing high-performance extremely low-bit neural networks.

## 5. Experimental Analysis

### 5.1. Experimental settings

**Implementation details.** We evaluate our method on the ImageNet dataset [8]. If not specified, we follow the standard practice for quantized models [17, 10] on quantizing the weights and activation for all convolution layers except the first convolution layer, last linear layer, and the convolution layers in the SE modules [14]. To fairly compare with quantized compact models, the FLOPs and BitOPs are defined as follows. Denote the FLOPs of the FP layer by $a$, the BitOPs of $m$-bit weight and $n$-bit activation quantized layer is $mn \times a$ following [41, 20, 29, 3, 36]. Further details can be found in Appendix.

**Search space.** Our search space is based on MobileNetV2 (MBV2) [30] and MobileNetV3 (MBV3) [13], which has the flexible input resolution, filter kernel size, depth (number of blocks in each stage), and width (number of channels). Our search space consists of multiple stages. Each stage stacks several inverted residual blocks. Further details about search space can be found in the Appendix. Unless otherwise noted, all results are sampled from the MBV3 search space denoted as OQAT, OQAT-MBV2 represents the MBV2 search space. Further details can be found in Appendix.

Table 4. The comparison of the search cost and retrain cost with existing quantization-aware NAS methods. $N$ denotes the number of models to be deployed. The total cost is calculated with $N = 40$.

| Methods | SPOS [11] | BMobi [29] | BATS [3] | APQ [35] | **OQAT** |
|---|---|---|---|---|---|
| search cost (GPU hours) | $288 + 24N$ | $29N$ | $6N$ | $2400 + 0.5N$ | **1200+0.5N** |
| retrain cost (GPU hours) | $240N$ | $256N$ | $75N$ | $30N$ | **0** |
| total cost (GPU hours) | $10.8k$ | $11.4k$ | $3.2k$ | $3.6k$ | **1.2k** |

Table 5. The best architecture for full-precision (FP) is not the best for quantization and bit inheritance can effectively improve quantization accuracy even compared with long-time QAT. The two architectures share similar FLOPS and FP accuracies (columns 2, 3) but show different quantization friendliness. And even we finetune the model with a complete QAT for 150/500 epochs (column 5, 6), the accuracy is still far from that of the subnet directly sampled from BI 2-bit supernet (column 4).

| Architecture | FLOPs | Top-1 Acc.(%) In FP Supernet | Top-1 Acc.(%) In BI 2-bit Supernet | Top-1 Acc.(%) QAT@150 | Top-1 Acc.(%) QAT@500 |
|---|---|---|---|---|---|
| Pareto model from FP supernet | 144 | 73.6% | 54.4% | 28.1% | 43.5% |
| Pareto model from 2-bit supernet | 142 | 73.4% | **56.1**% | 33.2% | 47.2% |

**Architecture search of quantized supernet.** For quantization-friendly analysis, we directly evaluate the sampled subnets from the supernet without further retraining. We randomly sample 10K candidate architectures from the supernet with the FLOPs of the corresponding floating-point models ranging from 50M to 300M (2K in every 50M interval). The no-retraining property enables us to conduct the analysis within a huge search space. For one specific model deployment requirement, we exploit a coarse-to-fine architecture selection procedure, similar to [38]. We first randomly sample 500 candidate architectures from the supernet within the FLOPs range of $\pm 10\%$. After obtaining the good skeletons (input resolution, depth, width) in the pareto front, we randomly perturb the kernel sizes to further search for better architectures.

## 5.2. The efficiency of OQAT

In Table 4, we compare the search cost and the retrain cost of OQAT with existing methods. The search cost is defined as the time cost of the supernet training and the search process to get the final $N$ searched models under latency targets. The retrain cost is defined as the training cost to get the final accuracy of the searched architecture. When $N$ is larger than 5, the retrain cost of SPOS [11] and BMobi [29] will surpass the total cost of OQAT. Compared with APQ [35], our method can reduce half of the total cost. APQ needs to train an FP supernet and sample thousands of FP subnets to perform quantization-aware training, and thus requires the transfer learning from the floating-point predictor to the quantization predictor. Our OQAT only needs to train one quantized supernet and support a huge search space with over $10^{20}$ subnets that can be directly sampled from supernet without retraining. Thus, the average computational cost is relatively low.

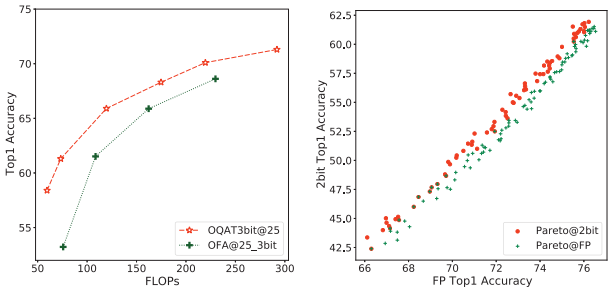
## 5.3. Ablation study

In Table 5, we validate the advantages of joint training and bit inheritance. The model in the first row is selected from the pareto front with the floating-point accuracy which corresponds to NAS-then-Quantize, and the latter is with the 2-bit accuracy as OQAT. With similar FLOPs, similar floating-point accuracy, OQAT surpasses the accuracy of Nas-then-Quantize with $1.7\%$ in 2-bit accuracy. We also perform QAT with 150 epochs and the joint training results in over $5\%$ accuracy improvement, which verifies the effectiveness of OQAT in finding quantization-friendly architectures. Although 2-bit models benefit from QAT with more epochs (e.g., 500 epochs), the achieved accuracy is still far from that directly sampled from the bit inheritance (BI) supernet. The huge accuracy improvement verifies that bit inheritance is a better practice compared with the existing routine of quantization-aware training because it alleviates the problem that quantized compact models with low bit-width are highly unstable to train.

## 5.4. Comparison with existing architectures

Benefiting from joint quantization and NAS with a large search space, as well as the bit inheritance for low-bit quantized supernet, we get high-performance OQATNets under extremely low bit-width. As shown in Figure 1(a) and Figure 1(b), OQATNets can be directly deployed for its superior accuracy, while the accuracy can be further improved by finetuning with 25 epochs denoted by OQAT@25.

We implement LSQ [9] denoted by LSQ* to construct strong baseline and compare with another strong quantization methods LSQ+ [2]. Our OQATNets outperforms multiple architectures like MobileNetV2 [30], EfficientNet-B0 [33] and MobileNetV3 [13] under all bit-widths we implements. **3 bit:** Our OQAT-3bit-L can also outperforms the accuracy of EfficientB0 by 1.3% with 15% FLOPs. **2 bit:** Our OQAT-2bit-M requires less FLOPs but achieves significantly higher Top-1 accuracy (61.7%) when compared with MobileNetV3@LSQ* (52.8%) and MobileNetV2@LSQ* (55.7%). The results verify that the OQAT results in quantization-friendly compact models.




(a) NAS-then-Quantize. (b) Pareto models.

Figure 4. Comparison of the parato models of NAS-then-Quantize and OQAT. The FLOPs of the corresponding floating-point architectures are used. The accuracy of Subnets@FP/3bit/2bit and Pareto@FP/3bit/2bit is obtained in the correponding FP/3bit/2bit supernet.

**NAS-then-Quantize.** We further compare with searched model produced by NAS-then-Quantize procedure (Figure 2(a)). As shown in Figure 4(a), we quantize the pareto models of the OFA[5] floating-point supernet and compare it with the pareto models of our OQATNets. In the comparison under 3 bit, the pareto curve of our OQAT is far above that of the NAS-then-Quantize.

In Figure 4(b), we sample 10k subnets from the search space, and we validate these architectures from the FP supernet and 2-bit supernet. The pareto front of the subnets denoted as Parato@FP are selected with the FP accuracy and Pareto@2bit are selected with the 2-bit accuracy. Pareto models as those models on the pareto front of the cost/accuracy trade-off curve. With the same accuracy of the floating-point models, the accuracy of the model from the 2-bit pareto is higher than the model from the FP pareto. If our target is to search architectures for the quantized models, searching from the quantized supernet as our OQAT did is better than searching from FP supernet and then quantized.

**Existing Quantization-aware NAS.** In Table 6, we compare our OQATNets model family with several searched models from existing quantization-aware NAS methods, named HAQ [34], SPOS [11], BMobi [29], BATS [3] and APQ [35].

Our OQATNets model family shows great advantages over existing weight-sharing methods corresponding to the paradigm of Figure 2(b). While SPOS [11] focuses on the search of network channels and bit-width of heavy ResNet [12], we focus on the search of compact models with fixed bit-width and achieve better results with fewer FLOPs. BMobi [29] and BATS [3] did not provide the results for 2-bit, 3-bit or 4-bit. Therefore, we would like not to directly compare our approach with BMobi and BATS, because the results are obtained from different bit-widths.

Table 6. Quantization-aware NAS performance under different bit-widths on ImageNet dataset. *Bit (W/A)* denotes the bit-width for both weights and activation. The number of bit for different layers is different for SPOS [11] and APQ [35]. BMobi [29], BATS [3], and OQAT use the same bit-width for different layers. L and M are short for for Large and medium model size.

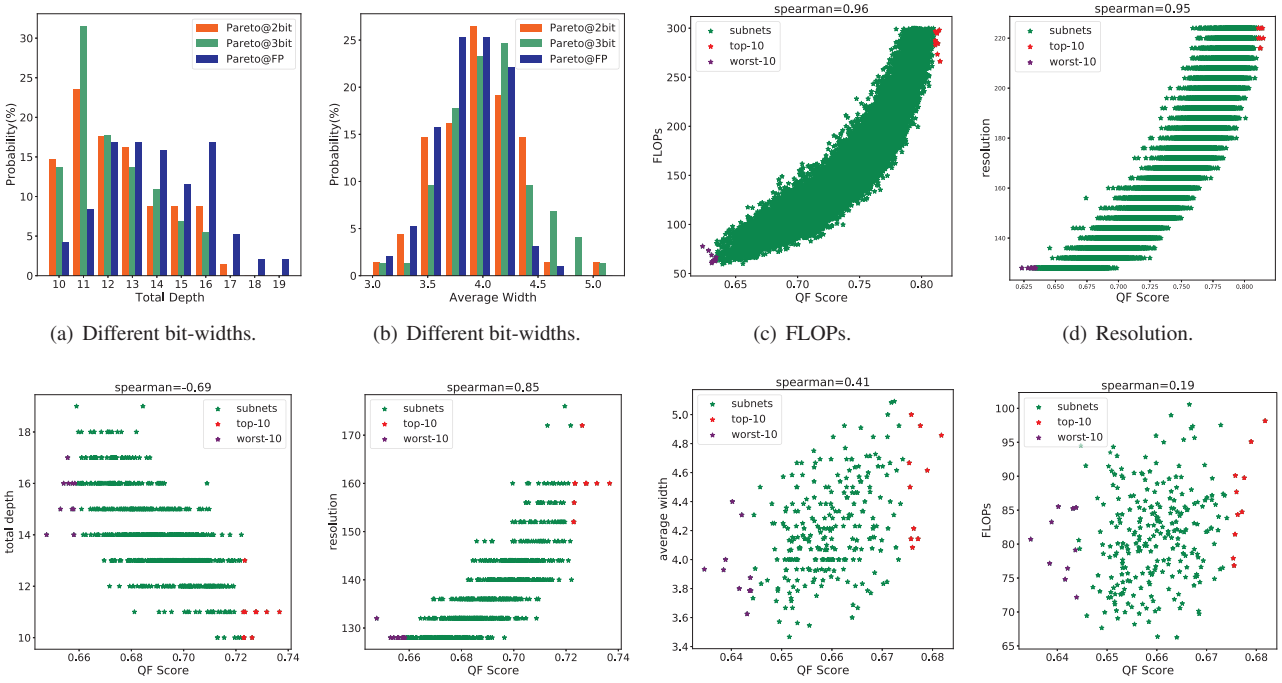
| Models | Bit(W / A) | BitOPs(G) | Top-1 |
|---|---|---|---|
| SPOS-ResNet34 | {1, 2, 3, 4} | 13.11 | 71.5 |
| SPOS-ResNet18 | {1, 2, 3, 4} | 6.21 | 66.4% |
| BATS-2× | 1 | 9.92 | 66.1% |
| BATS-1× | 1 | 6.30 | 60.4 % |
| BMobi-M1 | 1 | 3.97 | 59.3% |
| BMobi-M2 | 1 | 2.11 | 51.1% |
| **OQAT-3bit-L** | 3 | 3.07 | **71.3%** |
| **OQAT-3bit-M** | 3 | 1.92 | **68.3%** |
| **OQAT-2bit-M** | 2 | 1.21 | **61.7%** |
| APQ-B | {4, 6, 8} | 16.5 | 74.1% |
| APQ-A | {4, 6, 8} | 13.2 | 72.1% |
| **OQAT-MBV2-4bit-L** | 4 | 9.28 | **74.1%** |
| **OQAT-MBV2-4bit-M** | 4 | 6.85 | **72.4%** |

However, if only the FLOPs-accuracy trade-off is concerned, our OQAT with a higher bit-width can be a better solution. APQ utilizes transfer learning from FP predictor to quantized predictor which may bring proxy problems. Our OQAT-MBV2-4bit-L uses 43.7% less computation cost while maintaining the same accuracy as APQ-B.

### 5.5. Quantization-friendly architecture analysis.

With the large search space and numerous subnets which can be directly deployed, we can analyze the quantization-friendly architectures (QFA) under different bit widths. We sample 20k models from the search space with the flops in the range of $[50M, 300M]$, and we evaluate these architectures from the corresponding supernet to get the accuracy under different bit widths. To evaluate the quantization impact on different architectures, we define a quantization friendliness factor to evaluate QFA under different settings. It is calculated as the ratio $\mathbf{QF_k} = \frac{\mathbf{Acc_k}}{\mathbf{Acc_{FP}}}$ between the quantization accuracy under $k$ bit $\mathbf{Acc_k}$ and floating-point accuracy $\mathbf{Acc_{FP}}$. Spearman correlation coefficient (Spearman) is calculated to measure the correlation between $\mathbf{QF}$ score and elements like flops, resolution, total depth, average width(total width/total depth), and average kernel size(total kernel size/total depth). Only a part of the conclusion is listed as follows.

**QFA with different bit-widths.** We analyze the pareto models in the flops and accuracy curve under different bit widths. The distribution of depth, width reveals that the 2-bit models favor shallower depth as shown in Figure 5(a). There is no obvious favored pattern in terms of average width as shown in Figure 5(b).




(a) Different bit-widths. (b) Different bit-widths. (c) FLOPs. (d) Resolution.

(e) Given computation budget. (f) Given computation budget. (g) Given floating-point accuracy. (h) Given floating-point accuracy.

Figure 5. Quantization friendly architecture analysis. The quantization score of (b-h) is calculated with the 2bit quantization accuracy. Top-10 and worst-10 are selected with the highest or worst QF scores.

**Resolution is important for QFA.** In the visualization of sampled subnets and QF score under 2-bit, **QF** score increases greatly from 0.4 to 0.8 as the flops increases as shown in Figure 5(c). Among all the elements affecting FLOPs, we show that architectures with the large resolution are less sensitive to quantization as shown in Figure 5(d), which means that if several architectures with similar flops perform the same in floating-point accuracy, these architectures with large resolution tend to performs better in quantization.

**QFA with given computation budget.** We further compare the quantization-friendly pattern with a given computation budget, such as 200M FLOPs (the difference $\leq 3\%$). We plot 700 architectures in Figure 5(e) and 5(f). It is easy to conclude that the **QF** score is negatively correlated to the network depth and positively correlated to the input resolution. The **QF** score is almost irrelevant to the average width. Therefore the top-10 models with the highest **QF** score have shallower depth than the worst-10 models. It means that with a similar computation budget, we need to design QNN with shallower depth and large resolution.

**QFA with the same floating-point accuracy.** Under the same floating-point accuracy $\mathbf{Acc_{FP}}$, the quantization accuracy is decided by **QF** score. We plot over 200 architectures with floating-point accuracy $67.8\%$ with (the difference $\leq 0.2\%$). We find that the average width has a positive correlation with the QF score as shown in Figure 5(g). Besides, the quantization accuracy is less relevant to the FLOPs of models as shown in Figure 5(h). Therefore, in this case, choosing the wider models results in higher quantization accuracy.

## 6. Conclusion

In this paper, we present Once Quantization-aware Training (OQAT), a framework that deploys the searched quantized models without additional retraining and solves the problem of large accuracy degradation under ultra-low bit widths. With our proposed methods, we can search for the OQATNets model family which far exceeds architectures. Our results reveal the potential of high-performance extremely low-bit neural networks. A comprehensive study reveals the quantization-friendly architectures under different bit widths which might shed a light on further research of high performance extremely low-bit models.

## 7. Acknowledgement

This work was supported by the Australian Research Council Grant DP200103223, and the Australian Medical Research Future Fund MRFAI000085.

## References


[1] Milad Alizadeh, Arash Behboodi, Mart van Baalen, Christos Louizos, Tijmen Blankevoort, and Max Welling. Gradient l1 regularization for quantization robustness. *arXiv preprint arXiv:2002.07520*, 2020. 4

[2] Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. *arXiv preprint arXiv:2004.09576*, 2020. 1, 3, 5, 6

[3] Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. Bats: Binary architecture search. *arXiv preprint arXiv:2003.01711*, 2020. 1, 3, 4, 5, 6, 7

[4] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv preprint arXiv:1909.13863*, 2019. 1

[5] Han Cai, Chuang Gan, and Song Han. Once for all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019. 2, 3, 4, 7

[6] Hsin-Pai Cheng, Feng Liang, Meng Li, Bowen Cheng, Feng Yan, Hai Li, Vikas Chandra, and Yiran Chen. Scalenas: One-shot learning of scale-aware representations for visual recognition. *arXiv preprint arXiv:2011.14584*, 2020. 2

[7] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. 3

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[9] 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, 2, 3, 4, 6

[10] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4852–4861, 2019. 1, 5

[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. 1, 3, 6, 7

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3, 7

[13] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1314–1324, 2019. 3, 5, 6

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 5

[15] Peng Hu, Xi Peng, Hongyuan Zhu, Mohamed M Sabry Aly, and Jie Lin. Opq: Compressing deep neural networks with one-shot pruning-quantization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7780–7788, 2021. 1

[16] Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bit-widths. *arXiv preprint arXiv:1912.09666*, 2019. 4

[17] Jangho Kim, Yash Bhalgat, Jinwon Lee, Chirag Patel, and Nojun Kwak. Qkd: Quantization-aware knowledge distillation. *arXiv preprint arXiv:1911.12491*, 2019. 5

[18] Chuming Li, Xin Yuan, Chen Lin, Minghao Guo, Wei Wu, Junjie Yan, and Wanli Ouyang. Am-lfs: Automl for loss function search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8410–8419, 2019. 3

[19] Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Improving one-shot nas by suppressing the posterior fading. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13836–13845, 2020. 3

[20] Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. *arXiv preprint arXiv:1909.13144*, 2019. 1, 5

[21] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021. 1, 3

[22] Yuhang Li, Mingzhu Shen, Jian Ma, Yan Ren, Mingxin Zhao, Qi Zhang, Ruihao Gong, Fengwei Yu, and Junjie Yan. MQBench: Towards reproducible and deployable model quantization benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. 3

[23] Yuhang Li, Wei Wang, Haoli Bai, Ruihao Gong, Xin Dong, and Fengwei Yu. Efficient bitwidth search for practical mixed precision neural network. *arXiv preprint arXiv:2003.07577*, 2020. 3

[24] Feng Liang, Chen Lin, Ronghao Guo, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Computation reallocation for object detection. *arXiv preprint arXiv:1912.11234*, 2019. 3

[25] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 3

[26] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 3

[27] Asit Mishra, Eriko Nurvitadhi, Jeffrey J Cook, and Debbie Marr. Wrpn: wide reduced-precision networks. *arXiv preprint arXiv:1709.01134*, 2017. 1, 3

[28] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International*

*Conference on Machine Learning*, pages 7197–7206. PMLR, 2020. 1, 3

[29] Hai Phan, Zechun Liu, Dang Huynh, Marios Savvides, Kwang-Ting Cheng, and Zhiqiang Shen. Binarizing mobilenet via evolution-based searching. *arXiv preprint arXiv:2005.06305*, 2020. 3, 5, 6, 7

[30] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 5, 6

[31] Mingzhu Shen, Kai Han, Chunjing Xu, and Yunhe Wang. Searching for accurate binary neural architectures. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 1, 3

[32] Moran Shkolnik, Brian Chmiel, Ron Banner, Gil Shomron, Yuri Nahshan, Alex Bronstein, and Uri Weiser. Robust quantization: One model to rule them all. *arXiv preprint arXiv:2002.07686*, 2020. 4

[33] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019. 6

[34] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8612–8620, 2019. 1, 3, 7

[35] Tianzhe Wang, Kuan Wang, Han Cai, Ji Lin, Zhijian Liu, Hanrui Wang, Yujun Lin, and Song Han. Apq: Joint search for network architecture, pruning and quantization policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2078–2087, 2020. 1, 2, 3, 6, 7

[36] 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. 5

[37] Jiahui Yu and Thomas S Huang. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1803–1811, 2019. 2, 4

[38] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. *arXiv preprint arXiv:2003.11142*, 2020. 2, 3, 4, 6

[39] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. *arXiv preprint arXiv:1812.08928*, 2018. 4

[40] Dongzhan Zhou, Xinchi Zhou, Wenwei Zhang, Chen Change Loy, Shuai Yi, Xuesen Zhang, and Wanli Ouyang. Econas: Finding proxies for economical neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11396–11404, 2020. 3

[41] 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, 5