ONCE QUANTIZED FOR ALL: PROGRESSIVELY SEARCHING FOR QUANTIZED EFFICIENT MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Automatic search of Quantized Neural Networks has attracted a lot of attention. However, the existing quantization aware Neural Architecture Search (NAS) approaches inherit a two-stage search-retrain schema, which is not only time-consuming but also adversely affected by the unreliable ranking of architectures during the search. To avoid the undesirable effect of the search-retrain schema, we present Once Quantized for All (OQA), a novel framework that searches for quantized efficient models and deploys their quantized weights at the same time without additional post-process. While supporting a huge architecture search space, our OQA can produce a series of ultra-low bit-width(*e.g.* 4/3/2 bit) quantized efficient models. A progressive bit inheritance procedure is introduced to support ultra-low bit-width. Our discovered model family, OQANets, achieves a new state-of-the-art (SOTA) on quantized efficient models compared with various quantization methods and bit-widths. In particular, OQA2bit-L achieves 64.0% ImageNet Top-1 accuracy, outperforming its 2-bit counterpart EfficientNet-B0@QKD by a large margin of 14% using 30% less computation budget.

1 INTRODUCTION

Efficient architecture design (Sandler et al., 2018; Ma et al., 2018) and network quantization methods (Choi et al., 2018; Kim et al., 2019; Esser et al., 2019) are two promising research directions to deploy deep neural networks on mobile devices. Network quantization aims at reducing the number of bits for representing network parameters and features. On the other hand, Neural Architecture Search(NAS) (Howard et al., 2019; Cai et al., 2019; Yu et al., 2020) is proposed to automatically search for efficient architectures, which avoids expert efforts and design trials. In this work, we consider both of their advantages and explore NAS's ability in finding quantized efficient models.

Recent quantization methods quantize models with high floating-point performance no matter it is manually designed like MobileNetV2 (Sandler et al., 2018) or searched like EfficientNet-B0 (Tan & Le, 2019). However, the accuracy rank among floating-point models would change after they are quantized. The NAS-then-Quantize routine in Figure 1a that searches the best floating-point models and then quantizes the network with retraining, may fail to get a good quantized model. Directly using quantized models' performance to search seems to be a solution.

Existing quantization-aware NAS methods (Wang et al., 2019; Shen et al., 2019; Bulat et al., 2020; Guo et al., 2019; Wang et al., 2020) utilize a two-stage search-retrain schema as shown in Figure 1b. Specifically, they first search for one architecture under one bit-width setting[1], and then retrain the model under the given bit-width. This two-stage procedure will undesirably increase the number of models to be retrained if we have multiple deployment constraints and hardware bit-widths. Furthermore, due to the instability brought by quantization training, simply combining NAS and quantization will result in unreliable ranking (Li et al., 2019a; Guo et al., 2019) and sub-optimal quantized models (Bulat et al., 2020). Moreover, when the quantization bit is lower than 3 bits, the traditional training process is highly unstable and introduces very large accuracy degradation.

To alleviate the aforementioned problems, we present Once Quantized for All (OQA), a novel framework that: 1) searches for quantized network architectures and deploys their quantized weights

[1]One bit-width setting refers to a specific bit-width for each layer, where different layers could have different bit-widths.

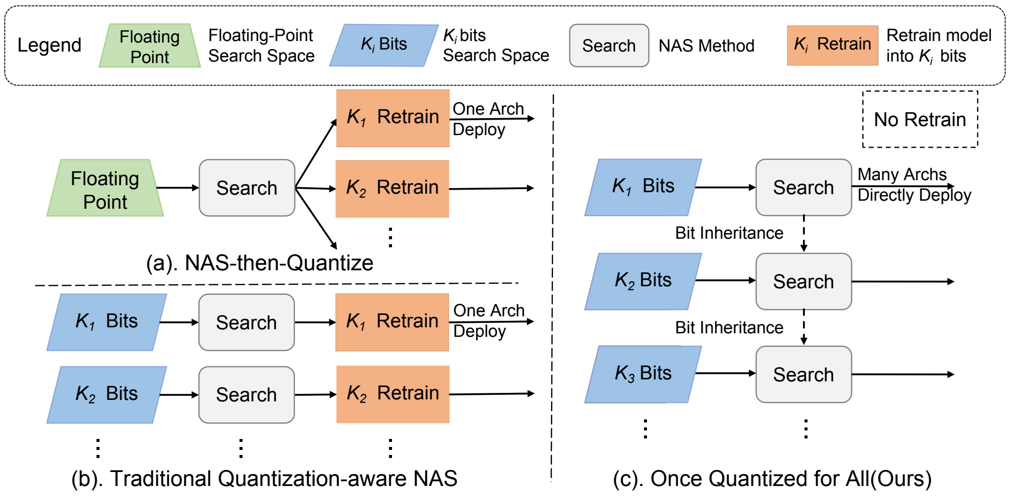


Figure 1: The overall frameworks of existing works on combining quantization and NAS and our method. (a) denotes directly converting the best searched floating-point architecture to quantization. (b) first adopts a QNN search algorithm to find a single architecture, then retrain the weights to quantization. Our OQA (c) can search for many quantized efficient models under various bit-width and deploy their quantized weights directly.

concurrently without retraining, 2) progressively produces a series of ultra-low bits(*e.g.* 4/3/2 bit) quantized models. Our approach leverages the recent NAS approaches which require no retraining (Yu & Huang, 2019; Cai et al., 2019; Yu et al., 2020). We adopt the search for kernel size, depth, width, and resolution in our search space. And we further enable searching them under different quantization bit-widths. Furthermore, we propose the bit inheritance mechanism to reduce the bit-width progressively so that we can utilize the higher bit models to provide better initialization and distillation knowledge for lower bit models. Benefiting from the no retraining property and large search space under different bit-width, we are able to evaluate the effect of network factors which could provide some insight into quantization-friendly architecture design.

Extensive experiments show the effectiveness of our approach. Our discovered quantized model family, OQANets, achieves state-of-the-art (SOTA) results on the ImageNet dataset under various quantization methods and bit-widths. In particular, our OQA2bit-L far exceeds the accuracy of 2bit Efficient-B0@QKD (Kim et al., 2019) by a large 14% margin using 30% less computation budget. Comparing with the quantization-aware NAS method APQ (Wang et al., 2020), our OQA4bit-L-MBV2 uses 43.7% less computation cost while maintaining the same accuracy as APQ-B.

To summarize, the contributions of our paper are three-fold:

- Our OQA is the first quantization NAS framework to search for the architecture of quantized networks and deploy their quantized weights without retraining.
- We present the bit inheritance mechanism to reduce the bit-width progressively so that the higher-bit models guide the search and training of lower-bit models.
- We provide some insight into quantization friendly architecture design. Our systematical analysis reveals that shallow-fat models are more likely to be quantization friendly than deep-slim models under low bit-width.

2 Related Work

Network Architecture Search without retraining Slimmable neural networks (Yu et al., 2018) first proposes to train a model to support multiple width multipliers(*e.g.* 4 different global width multipliers for MobileNetV2). OFA (Cai et al., 2019) and BigNAS (Yu et al., 2020) push the envelope forward in network architecture search(NAS) by introducing diverse architecture space (stage depth, channel width, kernel size, and input resolution). These methods propose to train a single over-parameterized supernet from which we can directly sample or slice different candidate architectures for instant inference and deployment. However, all of the aforementioned methods are tailored towards searching for floating-point efficient models. Converting the best floating-point architecture to quantization tends to result in sub-optimum quantization models. In the quantization area, recent papers (Jin et al., 2019a; Yu et al., 2019) propose to train a single model that can support different bit-width. But they only quantize manually design networks(*e.g.* ResNet, MobileNetV2) under relatively high bit-width (*e.g.* 4bit for MobileNetV2), our OQA can search architectures and produce lower bit-width(*e.g.* 2bit) quantized efficient models.

Quantization-aware Network Architecture Search Recent studies combine network quantization and NAS to automatically search for layer bit-width with given architectures or operations with given bit-width. HAQ (Wang et al., 2019) focuses on searching for different numbers of bits 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 (Shen et al., 2019) utilizes the genetic algorithm to search for network channels and BMobi (Phan et al., 2020) searches for the group number of different convolution layer under certain 1bit. SPOS (Guo et al., 2019) train a quantized one-shot supernet to search for bit-width and network channels for heavy ResNet (He et al., 2016). BATS (Bulat et al., 2020) devises a binary search space and incorporate it within the DARTS framework (Liu et al., 2018). The aforementioned methods concentrate on the quantization of heavy networks, like ResNet (He et al., 2016), or replace the depthwise convolution with group convolution. Moreover, they inherit a two-stage search-retrain schema: once the best-quantized architectures have been identified, they need to be retrained for deployment. This procedure will significantly increase the computational cost for the search if we have different deployment constraints and hardware bit-width. Compared with all these methods, our OQA can search for quantized efficient models and learn their quantized weights at the same time without additional retraining. Without our bit inheritance mechanism, these approaches also suffer from significant drop of accuracy when a network is quantized by very low bit like 2.

3 METHOD

3.1 OVERVIEW

Our OQA aims to obtain quantized efficient models which can directly be sampled from quantization supernet without retraining. As shown in Figure 1c, the overall procedure of OQA is as follows: Step 1, Quantized Supernet Training (Section 3.3): Train a K-bit supernet by learning the weight parameters and quantization parameters jointly. Step 2: given a constraint on computational complexity, search the architecture with the highest quantization performance on the validation dataset from the supernet. If $K = 2$, the whole process is finished. Step 3, Bit Inheritance (Section 3.4): 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.

The starting bit-width K of OQA and the steps of the bit-inheritance procedure can be arbitrary. In this paper, we focus on the efficient models under one fixed low bit-width quantization strategy, thus we define the bit-width is 4/3/2.

3.2 PRELIMINARIES

Neural Architecture Search without Retraining. Recently, several NAS methods (Yu et al., 2018; Cai et al., 2019; Yu et al., 2020) are proposed to directly obtain deployable subnets from a well-trained supernet without retrain. Specifically, a supernet with the largest possible depth (number of layers), width (number of channels), kernel size, and input resolution is trained. Then the subnet with top accuracy among the set of subnets satisfying a given computational complexity requirement is selected as the searched network. A subnet is obtained from parts of the supernet with depth, width, and/or kernel size smaller than the supernet. The subnet uses the well-trained parameters of the supernet for direct deployment without further retraining.

Quantization Network Learning. In the quantization neural networks, a quantization function turns the floating-point weights, and activations into integers of given bit-width in the forward pass.

Given bit-width K, activations are quantized into unsigned integers in the range of $[0, 2^K - 1]$ and weights are quantized into signed integers in the range of $[-2^{K-1}, 2^{K-1} - 1]$. To enable the training of quantized supernets, we choose a learnable quantization function following recent state-of-the-art quantization method LSQ (Esser et al., 2019). Given floating-point weights or activation $\mathbf{v}$, and learnable scale s, the quantization function Q and its corresponding approximate gradient using the straight through estimator (Bengio et al., 2013). Take the activation $\mathbf{v}$ as the example, we have

$$\begin{aligned} &\text{Quantization function: } \mathbf{v}^q = Q(\mathbf{v}, s) = \lfloor \text{clip}(\frac{\mathbf{v}}{|s|}, Q_{min}, Q_{max}) \rceil \times |s|, \\ &\text{Approximate gradient: } \frac{\partial Q(\mathbf{v})}{\partial \mathbf{v}} \approx \mathbb{I}(\frac{\mathbf{v}}{|s|}, Q_{min}, Q_{max}), \end{aligned} \tag{1}$$

where all operations for $\mathbf{v}$ are element-wise operations, $clip(z, r1, r2)$ returns z with values below $r1$ set to $r1$ and values above $r2$ set to $r2$, $\lfloor z \rceil$ rounds z to the nearest integer, Q_{min} and Q_{max} are, respectively minimum and maximum integers for the given bit-width k, $\mathbb{I}(\frac{\mathbf{v}}{|s|}, Q_{min}, Q_{max})$ means the gradient of $\mathbf{v}$ in the range of $(Q_{min} \times |s|, Q_{max} \times |s|)$ is approximated by 1, otherwise 0. $|s|$ is the absolute value of s, ensuring that the semantics of scale s is only interval, without inverting the signs of weights or activation. The scale s is learned by back-propagation and initialized as $\frac{2}{\sqrt{Q_{max}}} \times |\bar{\mathbf{v}}|$, where $|\bar{\mathbf{v}}|$ denotes the mean of $|\mathbf{v}|$.

3.3 Quantized NAS without Retraining

Existing problems of Quantization NAS. Existing weight-sharing-based quantization NAS methods suffer from more unreliable order preserving (Guo et al., 2019; Bulat et al., 2020), as the quantization function introduces more instability on the learned weights. In our perspective, combining non-retrain NAS methods with quantization to avoid this problem can be a natural solution.

Search space and quantized supernet. We use a search space based on MobileNetV3 (Howard et al., 2019) and MobileNetV2 (Sandler et al., 2018), 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 Appendix A.4, A.5

Unlike the floating-point supernet training (Cai et al., 2019; Yu et al., 2020), we utilize the quantization function as Eq. 1 to discretize the weights and activation values for the quantization supernet training. Meanwhile, the floating-point weights need to be retained for reducing quantization loss at the training stage of each bit-width. For weight $\mathbf{w}$ and input activation $\mathbf{a}$ of a convolution layer, we define the corresponding learnable scales of activation and weight as $s_{\mathbf{a}}$ and $s_{\mathbf{w}}$. The forward pass for a quantized convolution layer is defined as follows:

$$\begin{aligned} \mathbf{w}^q &= Q(\mathbf{w}, s_{\mathbf{w}}), \\ \mathbf{a}^q &= Q(\mathbf{a}, s_{\mathbf{a}}), \\ \mathbf{y} &= \mathbf{w}^q * \mathbf{a}^q, \end{aligned} \tag{2}$$

where $Q(\cdot, \cdot)$ is the learnable quantization function defined in Eq. 1, * is the convolution operation and $\mathbf{y}$ is the output of this layer.

Subnet sampling. During the supernet training, different subnets are sampled and trained in each iteration. In non-retrain NAS methods, a subnet has a smaller scale than the supernet in resolution, width, depth, and kernel size, and is obtained by cropping the corresponding part from the supernet. In our settings, a stage with d blocks in a subnet inherits the weights from the first d blocks in the same stage in supernet. A depthwise convolution layer in a subnet with width e and kernel size k are cropped from the central $k * k$ region of the first e kernels in the supernet's corresponding convolution layer. The input image of each subnet are resized to its resolution r.

Our subnet sampling strategy combines the supernet training pipeline proposed in (Cai et al., 2019; Yu et al., 2020), and it has two stages. We first only sample the biggest quantized subnet until it converges as in (Cai et al., 2019). Afterward, we use sandwich rules to sample subnets, which shows efficiency in (Yu & Huang, 2019). Further details can be found in the Appendix A.2.

Architecture search of quantized supernet. We directly evaluate the sampled subnets from the supernet without further retraining. It's worth to mention that we use the predictive accuracy on 10K validation images sampled from *trainset* to measure the subnets in the search procedure. Furthermore, we exploit a coarse-to-fine architecture selection procedure, similar to Yu et al. (2020). We first 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). After obtaining the good skeletons (input resolution, depth, width) in the pareto front of the first 10K models, we randomly perturb the kernel sizes to further search for better architectures.

3.4 Quantization NAS with Bit Inheritance

The problem of quantization NAS with lower bits. When the quantization bit is lower than 3 bits, the traditional quantization-aware training (QAT) (Kim et al., 2019; Bhalgat et al., 2020) process is highly unstable and introduces very large accuracy degradation for the challenging case of the 2 bit-width model. Using the approach introduced in Section 3.3, we obtain the quantized supernet with the highest $K = 4$ bit-width. To further obtain the quantized supernets of lower bit-widths (*e.g.* $K - 1$, $K - 2$), we can use the QAT to directly train quantized supernets for each bit-width. As shown in Table 1, the biggest architecture in our search space suffers a 19.1% accuracy drop between 4-bit and 2-bit when using the QAT. Besides, QAT requires much more computational cost.

Inheritance from the high bit to low bit. We propose a bit inheritance procedure to compensate for both the disadvantages. First, we use the K bit-width supernet to provide a good initialization for the weights of $K - 1$ bit-width supernet. In the initialization for $K - 1$ bit-width supernet, we first inherit the weight parameters and scale parameters from K bit-width supernet. Then the scale parameters are doubled because they map the floating-point values to the integer range of $K-1$ bit-width, which is half the range of K bit-width. Finally, to compensate for the statistics shift of each layer's output caused by quantization error, we forward the model to recalculate the mean and variance of the BatchNorm layers (Yu & Huang, 2019) with randomly sampled 4096 training images. During training, we use the K and $K - 1$ bit-width supernets as teacher and student, and train them in a knowledge distillation way to further reduce the quantization error between the $K - 1$ and K bit-width parameters.

Table 1: The accuracy of the biggest model in quantization-aware training(QAT) and progressive bit inheritance from high bits to low bits. Start denotes train with one epoch, end denotes train at the end.

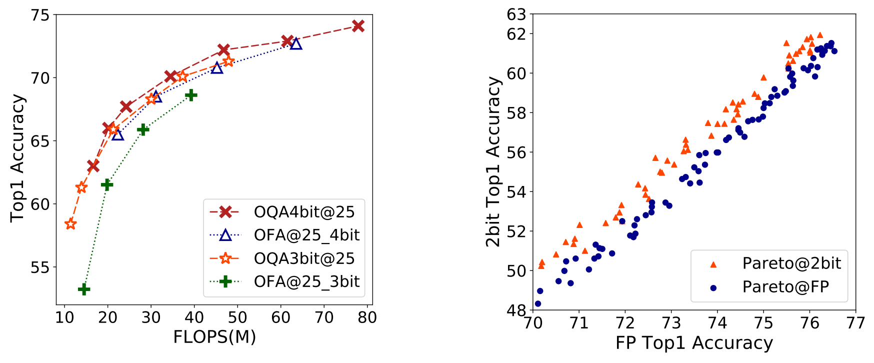
	4/4	3/3	2/2
QAT-Start	48.1%	23.2%	0.8%
QAT-End	75.1%	72.1%	56%
BitInheritance-Start	-	71.7%	49.9%
BitInheritance-End	-	72.7%	63.9%

The benefit of bit inheritance. In the Appendix A.1, we show that the loss of the $K-1$ bit network is bounded (close to that of the K bit network) if bit inheritance is used, where the parameters of the $K - 1$ bit network inherit from the parameters of the $K - 1$ bit network. Therefore, bit inheritance help to guarantee the 2-bit network to be close to the 3-bit network in training loss. In comparison, existing methods like QAT start from floating point network which is far away from the 2-bit network parameters and cause unstable training. The experimental results in Table 1 also validate the effectiveness of our design. The row named BitInheritance-Start means with only one epoch training, the initial accuracy is good enough in 3 bit. After finetuning with fewer epochs, the bit-inheritance achieves higher accuracy performance, especially for 2-bits.

4 Experimental Analysis

4.1 Experimental settings and implementation details

We evaluate our method on the ImageNet dataset (Deng et al., 2009). If not specified, we follow the standard practice for quantized networks (Kim et al., 2019; Gong et al., 2019) on quantizing



(a) (b)

Figure 2: Comparison of the parato models of NAS-then-Quantize and OQA. (a) OFA FP supernet is used for NAS and LSQ is used as the quantization method. @25 denotes finetuning for 25 epoch. (b) The accuracy of Pareto@2bit/FP is obtained in the correponding 2bit/FP supernet.

the weights and activation for all convolution layers except the first convolution, last linear layer, and the convolution layers in the SE modules(Hu et al., 2018). To fairly compare with quantized efficient models, the FLOPs is used and defined as follows. Denote the FLOPs of the FP layer by a, following (Zhou et al., 2016; Li et al., 2019b; Phan et al., 2020; Bulat et al., 2020), the FLOPs of the FP layer is a, and the FLOPs of m bit weight and n bit activation quantized layer is $\frac{mn}{64} \times a$. Unless otherwise noted, all results are sampled from MBV3 search space denoted as OQA, OQA-MBV2 represents the MBV2 search space. Further training details can be found in Appendix A.2.

4.2 NAS-THEN-QUANTIZE OR OQA

We denote pareto models as those models on the pareto front of cost/accuracy trade-off curve. In Figure 2a, we quantize the pareto models of the OFA floating-point supernet, which corresponds to the NAS-then-Quantize procedure in Figure 1a. We compare it with the pareto models of our OQA that directly obtained from the quantization supernet. In the comparison under 3bit, the pareto curve of our OQA is far above that of the NAS-then-Quantize.

In Figure 2b, we sample 10k subnets from the search space, and we validate these architectures from the FP supernet and 2bit supernet. The pareto front of the subnets denoted as Parato@FP are selected with the FP accuracy and Pareto@2bit are selected with the 2bit accuracy. With the same accuracy of the floating-point models, the accuracy of the model from the 2bit pareto is higher than the model from the FP pareto. If our target is to search architectures for the quantized models, Figure 2b shows that searching from the quantization supernet as our OQA did is better than searching from FP supernet and then quantize. The advantage is more evident for lower bits. Further details can be found in Appendix A.3.

4.3 EXISTING QUANTIZATION-AWARE NAS OR OQA

In Table 2, we compare our OQANet model family with several quantization-aware NAS methods, named SPOS (Guo et al., 2019), BMobi (Phan et al., 2020), BATS (Bulat et al., 2020) and APQ (Wang et al., 2020).

With the quantization supernets without retraining, our OQANet model family shows great advantages over traditional weight-sharing methods corresponding to the paradigm of Figure 1b. While SPOS (Guo et al., 2019) focuses on the search of network channels and bit-width of heavy ResNet (He et al., 2016), we focus on the search of efficient models with fixed bit-width and achieve better results with fewer FLOPs. BMobi (Phan et al., 2020) and BATS (Bulat et al., 2020) did not provide the results for 2bits, 3bits or 4 bits. Therefore, we would like not to directly compare our approach with BMobi and BATS, because the results are obtained from different bit-widths. However, if only the FLOPs-accuracy trade-off is concerned, our OQA with higher bit-width can be a better solution. APQ corresponds to the NAS-then-Quantize paradigm in Figure 1a. It first searches for floating-point network architecture in the FP supernet, then trains a quantization network predictor

Table 2: 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 (Guo et al., 2019) with bit-width in the range of $\{1, 2, 3, 4\}$ and APQ (Wang et al., 2020) with bit-width in the range of $\{4, 6, 8\}$. BMobi (Phan et al., 2020), BATS (Bulat et al., 2020), and OQA use the same bit-width for different layers.

Methods	Models	Bit (W / A)	FLOPs (M)	Top-1 Acc.(%)
SPOS	ResNet-34	$\{1, 2, 3, 4\}$	337	71.5
SPOS	ResNet-18	$\{1, 2, 3, 4\}$	221	66.4
BATS	2×	1	155	66.1
BATS	1×	1	98.5	60.4
BMobi	M2	1	62	59.3
BMobi	M3	1	33	51.1
OQA	OQA3bit-L	3	48	**71.3**
OQA	OQA3bit-M	3	30	**68.3**
OQA	OQA2bit-M	2	19	**61.7**
APQ	APQ-B	$\{4, 6, 8\}$	258	74.1
APQ	APQ-A	$\{4, 6, 8\}$	206	72.1
OQA	OQA4bit-L-MBV2	4	145	**74.1**
OQA	OQA4bit-M-MBV2	4	107	**72.4**

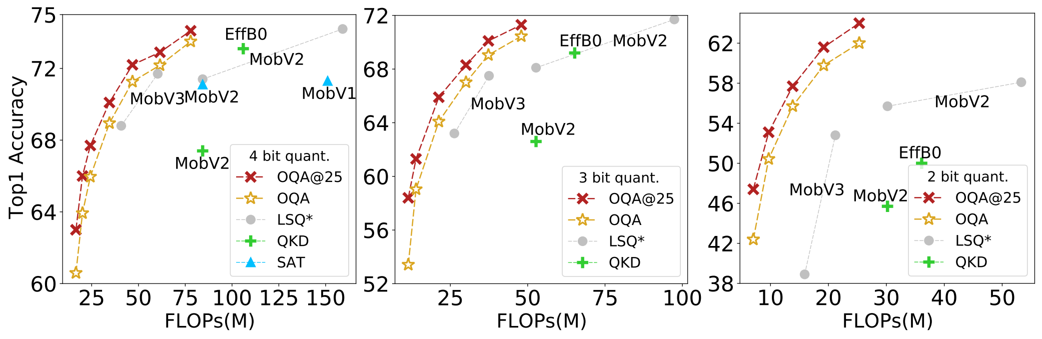


Figure 3: Comparison with the state-of-the-art quantization methods (LSQ , QKD, SAT) in various network(MobileNetV2/V3, EfficientNet-B0) on the ImageNet dataset.

to predict the searched quantized architecture. The transfer learning from FP predictor to quantization predictor brings the proxy problem and it also needs retraining. Our OQA4bit-L-MBV2 uses 43.7% less computation cost while maintaining the same accuracy as APQ-B.

4.4 Further comparison with NAS-then-quantize methods

We compare with several strong quantization methods including LSQ (Esser et al., 2019), LSQ+ (Bhalgat et al., 2020), APOT (Li et al., 2019b), QKD(Kim et al., 2019), SAT (Jin et al., 2019b), and LSQ* which is the LSQ implemented by us on different models to construct strong baselines. The result of 4bit ResNet-18@LSQ* validates that our implementation is comparable.

Our OQA benefits from joint quantization and network architecture search, as well as the bit inheritance for lower bits. As shown in Table 3, our OQANets outperforms multiple quantization methods on models like MobileNetV2 (Sandler et al., 2018), EfficientNet-B0 (Tan & Le, 2019) and MbV3 (Howard et al., 2019) under all bit-widths we implements. **4bits:** OQA4bit-L has 1% accuracy gain than Efficient-B0@QKD. OQA4bit-M outperforms ResNet-18@LSQ with 10% of its FLOPs. **3bits:** Our OQA3bit-L can also match the accuracy of 3bit ResNet-18@LSQ with 13% FLOPs and 3 bit Efficient-B0@QKD with 74% FLOPs. **2bits:** Our OQA2bit-L requires less FLOPs but achieves significantly higher Top-1 accuracy (64.0%) when compared with EfficientNet-B0@QKD (50.0%) and MobileNetV2@LSQ* (55.7%). The results verify that the joint search of quantization parameters and network architectures results in more quantization-friendly efficient models. We also show more searched models in the Figure 3 and our OQANets significantly outperform other quantization methods.

Table 3: ImageNet performance under 4, 3, 2 bit-width. OQA4bit-M and OQA4bit-L denote medium and large model size in the 4bit OQANets family respectively. @25 means we take weights from the supernet and finetune for 25 epochs. W/A denotes the bit-width for both weights and activation.

Models	Method	Bit (W / A)	FLOPs (M)	Top-1 Acc.(%)
Efficient-B0	QKD	4	106	73.1
OQA4bit-L	OQA@25	4	73	**74.1**
ResNet-18	LSQ / LSQ*	4	542 / 542	71.1 / 70.9
MobileNetV2	LSQ* / SAT	4	85	71.3 / 71.1
MbV3-L (1.0x)	LSQ*	4	60	71.7
OQA4bit-M	OQA@25	4	47	**72.3**
ResNet-18	LSQ / APOT	3	357 / 298	70.6 / 69.9
Efficient-B0	QKD	3	65	69.2
OQA3bit-L	OQA@25	3	48	**71.3**
MobileNetV2	LSQ* / QKD	3	53 / 53	68.2 / 62.6
MbV3-L 1.0x	LSQ*	3	38	67.5
OQA3bit-M	OQA@25	3	30	**68.3**
Efficient-B0	QKD / LSQ+	2	36	50.0 / 49.1
MobileNetV2	LSQ* / QKD	2	30	55.7 / 45.7
OQA2bit-L	OQA@25	2	25	**64.0**
OQA2bit-S	OQA@25	2	14	**57.7**

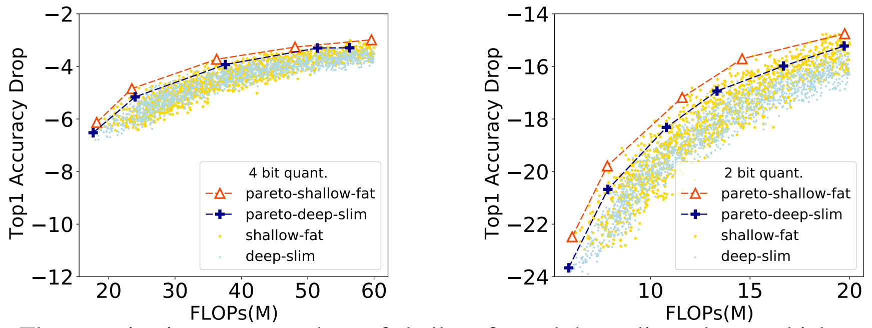


Figure 4: The quantization accuracy drop of shallow-fat and deep-slim subnets which are sampled from FP/4/2 bit supernets on the ImageNet dataset.

4.5 Shallow-fat or Deep-slim

With the different quantization supernet obtained by the bit inheritance process, we analyze the following factors of a network: depth(D), kernel size(K), expand width(E) and bits(B). According the intuitive shape, we divide the models into two group: 1) **shallow-fat:** D, K and E are sampled from $\{2, 3\}$, $\{5, 7\}$ and $\{4, 6\}$ respectively, 2) **deep-slim:** D, K, and E are sampled from $\{3, 4\}$, $\{3, 5\}$ and $\{3, 4\}$ respectively.

Following the rules above, we randomly generate 1.5K architectures and obtain the accuracy of the floating-point and quantized models from the corresponding supernet. The input resolution is also randomly sampled. For a certain model, we use the accuracy drop from floating-point to quantization to measure whether it is quantization friendly. As shown in Figure 4, the shallow-fat models are more quantization friendly than deep-slim models. When the 4-bit and 2-bit situations are compared, the trend becomes more obvious when the quantization bit is lower.

5 Conclusion

In this paper, we present Once Quantized for All (OQA), a novel framework that searches quantized neural networks (QNNs) under the bit-width of $4/3/2$ and deploys the learned quantized models at the same time without additional retraining. With our proposed methods, we can search for the OQANet model family which far exceeds the existing quantization aware nas and quantization methods which are applied to mobile models. Our results reveal the potential of ultra-low bit quantized models under mobile settings.

REFERENCES

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

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.

Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. Bats: Binary architecture search. *arXiv preprint arXiv:2003.01711*, 2020.

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.

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.

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*, pp. 248–255. Ieee, 2009.

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

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*, pp. 4852–4861, 2019.

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.

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*, pp. 770–778, 2016.

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*, pp. 1314–1324, 2019.

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

Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bitwidths. *arXiv preprint arXiv:1912.09666*, 2019a.

Qing Jin, Linjie Yang, and Zhenyu Liao. Towards efficient training for neural network quantization. *arXiv preprint arXiv:1912.10207*, 2019b.

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

Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Improving one-shot nas by suppressing the posterior fading. *arXiv preprint arXiv:1910.02543*, 2019a.

Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: A non-uniform discretization for neural networks. *arXiv preprint arXiv:1909.13144*, 2019b.

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

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 116–131, 2018.

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.

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*, pp. 4510–4520, 2018.

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*, pp. 0–0, 2019.

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

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*, pp. 8612–8620, 2019.

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*, pp. 2078–2087, 2020.

Haichao Yu, Haoxiang Li, Honghui Shi, Thomas S Huang, and Gang Hua. Any-precision deep neural networks. *arXiv preprint arXiv:1911.07346*, 2019.

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

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

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.

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.

A APPENDIX

A.1 EFFECTIVENESS OF BI

For each w_i, we have:

$$\begin{aligned} Q(\mathbf{w_i}, s_K) &= \lfloor \text{clip}(\frac{\mathbf{w}_i}{|s_K|}, -2^{K-1}, 2^{K-1}-1) \rceil \times |s_K|, \\ Q(\mathbf{w_i}, s_{K-1}) &= \lfloor \text{clip}(\frac{\mathbf{w}_i}{|s_{K-1}|}, -2^{K-2}, 2^{K-2}-1) \rceil \times |s_{K-1}| \\ &= 2\lfloor \text{clip}(\frac{\mathbf{w}_i}{|2s_K|}, -2^{K-2}, 2^{K-2}-1) \rceil \times |s_K|. \end{aligned} \tag{3}$$

Based on this expression, we further get:

$$\begin{aligned} &|Q(\mathbf{w_i}, s_K) - Q(\mathbf{w_i}, s_{K-1})| = \\ &|\lfloor \text{clip}(\frac{\mathbf{w}_i}{|s_K|}, -2^{K-1}, 2^{K-1}-1) \rceil - 2\lfloor \text{clip}(\frac{\mathbf{w}_i}{|2s_K|}, -2^{K-2}, 2^{K-2}-1) \rceil| \times |s_K|. \end{aligned} \tag{4}$$

For any $\mathbf{w_i}$ and s_K, we have:

$$|\lfloor \text{clip}(\frac{\mathbf{w_i}}{|s_K|}, -2^{K-1}, 2^{K-1}-1) \rceil - 2\lfloor \text{clip}(\frac{\mathbf{w_i}}{|2s_K|}, -2^{K-2}, 2^{K-2}-1) \rceil| \leq 1. \tag{5}$$

Thus,

$$\begin{aligned} |Q(\mathbf{w_i}, s_K) - Q(\mathbf{w_i}, s_{K-1})| &\leq |s_K|, \\ &\text{and} \\ ||Q(\mathbf{w}, s_K) - Q(\mathbf{w}, s_{K-1})||_1 &\leq N_w \cdot |s_K|. \end{aligned} \tag{6}$$

A.2 TRAINING DETAILS

Dataset config: We evaluate our method on the ImageNet dataset. The training dataset is made up of 1.28 million images with resolution 224×224 belonging to 1000 classes and the validation set has 50k images. For ImageNet training, we use the typical random resized crop, randomly horizontal flipping and color jitter of $[32/255, 0, 0.5, 0]$ for data augmentation. During evaluation, we first determine the active image size s, and resize the image into $\lceil s/0.875 \rceil \times \lceil s/0.875 \rceil$ and center crop $s \times s$ image.

Quantization aware training: We reimplement LSQ (Esser et al., 2019) as our base quantization method. We start from a floating-point model and finetune the model for 150 epochs. The optimizer is SGD with Nesterov momentum 0.9 and weight decay 3e-5, and the label smoothing ratio is 0.1. The initial learning rate is 0.04 under the batch of 1024, with the cosine annealing schedule. The dropout rate is 0.1. Except for learning rate and training epochs, we follow this protocol in the OQA procedure.

OQA procedure: Combining the advantages of OFA (Cai et al., 2019) and BigNas (Yu et al., 2020), the overall OQA procedure is divided into four steps as follow:

Step0: we train the 4bit biggest models in the search space. It follows the typical quantization-aware training, we use the floating-point pre-trained model as initialization and finetuning for 150 epoch. The learning rate is 0.04 with a batch-size of 1024.

Step1: in the supernet training phase, the biggest model obtained in step 0 is used as initialization. The input resolution, kernel size, width, and depth are random sampled. This whole process takes 200 epochs. In one iteration, four models are sampled, which is the biggest subnet and the smallest subnet and two random sampled subnets. The learning rate is 0.02 with a batch size of 1024.

With bit inheritance, the training of the 3/2bit supernet is simplified. And the training time is reduced.

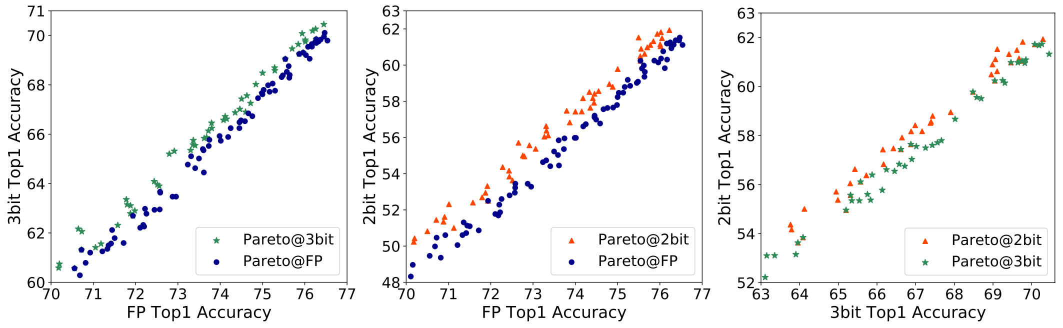


Figure 5: Comparison of the Top-1 validation accuracy on ImageNet dataset between the FP pareto, 3bit pareto, and 2bit pareto. The pareto is selected from the corresponding supernet and the accuracy is also obtained from the supernet.

Step2: In the 3bit supernet, we use the 4bit-supernet obtained in Architecture shrinking Step1 part2 as initialization. We directly random sample input resolution, kernel, width, and depth. four models are sampled, which is the biggest subnet and the smallest subnet and two random sampled subnets for one update. We only use 25 epochs and the learning rate is 0.0016.

Step3: In the 2bit supernet, we use the 3bit-supernet obtained in Step2 as initialization. We also directly random sample resolution, kernel, width, and depth. Four models are sampled, which is the biggest subnet and the smallest subnet and two random sampled subnets for one update. We only use 120 epochs and the learning rate is 0.0256.

OQA subnet finetuning: Our OQA performance can be further improved by finetuning the subnet weights sliced from the OQA supernet as suggested by OFA (Cai et al., 2019). The accuracy of the subnet is already higher than training from scratch. In default, the subnets are finetuned for 25epochs. The initial learning rate is 0.0016 with a batch size of 1024, with the cosine annealing schedule.

A.3 NAS-THEN-QUANTIZE OR OQA

In Figure 5, we sample 10k subnets from the search space, and we validate these subnets from the FP supernet, 3bit and 2bit supernet. The pareto front of the subnets denoted as Parato@FP are selected with the accuracy in the floating-point supernet and Pareto@3bit/Pareto@2bit are selected with the accuracy in the 3bit/2bit supernet. In Figure 5, the first two figure reveals that as the bit decreases, the accuracy gain increases with searching directly in the quantization supernet. We further compare the Pareto@3bit and Pareto@2bit, and it shows that with the same 3bit accuracy, the accuracy of the model from the 2bit pareto is higher than the model from the 3bit pareto, but the accuracy gain is less compared with the FP pareto. To search for 2bit quantized models, it is best to search directly in the 2bit supernet, and it is better to search in the 3bit supernet than searching in the FP supernet.

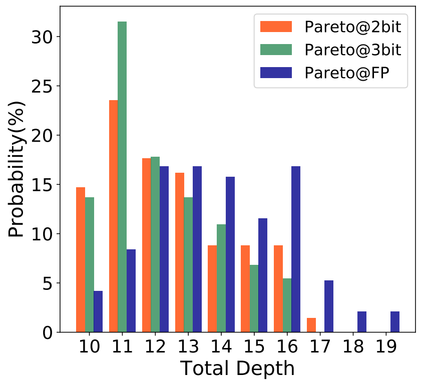


Figure 6: The probability distribution of total depth of the candidate architectures in the FP pareto,3bit pareto, and 2bit pareto.

Observed the difference in the accuracy of the pareto architectures in different supernets at the same 2bit, we are curious whether the accuracy performance is attributed to the structure difference in the pareto architecture. In Figure 6, we plot the distribution of the total depth of the pareto subnets from different supernet. In the depth dimension, the distribution reveals that Pareto@FP favors deeper models while Pareto@3bit/2bit favors shallow models. And the distribution of depth is closer between Pareto@3bit and Pareto@2bit.

A.4 MOBILENETV3 SEARCH SPACE

Table 4: MBConv refers to inverted residual block which has a '1×1 pointwise - $k \times k$ depthwise- 1×1 pointwise' structure without SE module (Hu et al., 2018), MBConv-SE is the MBConv block with SE module. Channels means the number of output channels in this stage. Depth means the number of blocks in this stage. Expand ratio refers the expand ratio of input channels which controls the width of the depthwise convolution. Convolution layers in the first and last has no expand ratio. Kernel size refers to the kernel size k of the depthwise convolution.

Stage	Operator	Resolution	Channels	Depth	Expand ratio	Kernel size
	Conv	128×128 - 224×224	16	1		1
1	MBConv	64×64 - 112×112	16	1	1	3
2	MBConv	64×64 - 112×112	24	2, 3, 4	3, 4, 6	3, 5, 7
3	MBConv-SE	32×32 - 56×56	40	2, 3, 4	3, 4, 6	3, 5, 7
4	MBConv	16×16 - 28×28	80	2, 3, 4	3, 4, 6	3, 5, 7
5	MBConv-SE	8×8 - 14×14	112	2, 3, 4	3, 4, 6	3, 5, 7
6	MBConv-SE	8×8 - 14×14	160	2, 3, 4	3, 4, 6	3, 5, 7
	Conv	4×4 - 7×7	960	1		1
	Conv	1×1	1280	1		1

A.5 MOBILENETV2 SEARCH SPACE

Table 5: MBConv refers to inverted residual block which has a '1×1 pointwise - $k \times k$ depthwise- 1×1 pointwise' structure without SE module (Hu et al., 2018), MBConv-SE is the MBConv block with SE module. Channels means the number of output channels in this stage. Depth means the number of blocks or layers in this stage. Expand ratio refers the expand ratio of input channels which controls the width of the depthwise convolution. Convolution layers in the first and last has no expand ratio. Kernel size refers to the kernel size k of the depthwise convolution.

Stage	Operator	Resolution	Channels	Depth	Expand ratio	Kernel size
	Conv	128×128 - 224×224	32	1		3
1	MBConv	64×64 - 112×112	16	1	1	3
2	MBConv	64×64 - 112×112	24	2, 3, 4	3, 4, 6	3, 5, 7
3	MBConv	32×32 - 56×56	40	2, 3, 4	3, 4, 6	3, 5, 7
4	MBConv	16×16 - 28×28	80	2, 3, 4	3, 4, 6	3, 5, 7
5	MBConv	8×8 - 14×14	96	2, 3, 4	3, 4, 6	3, 5, 7
6	MBConv	8×8 - 14×14	192	2, 3, 4	3, 4, 6	3, 5, 7
7	MBConv	4×4 - 7×7	320	1	3, 4, 6	3, 5, 7
	Conv	4×4 - 7×7	1280	1		1