SpaceEvo: Hardware-Friendly Search Space Design for Efficient INT8 Inference

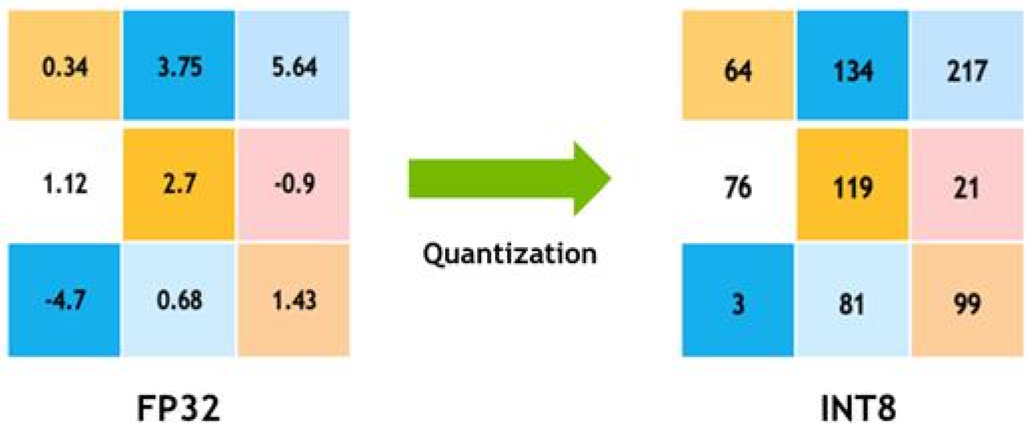
ICCV'23

Li Lyna Zhang[1*‡] Xudong Wang[2*§] Jiahang Xu[1] Quanlu Zhang[1] Yujing Wang[3]
Yuqing Yang[1] Ningxin Zheng[1] Ting Cao[1] Mao Yang[1]

From Microsoft Research Asia

Introduction

- INT8 quantization： compressing models by reducing the number of bits required to represent weights or activations

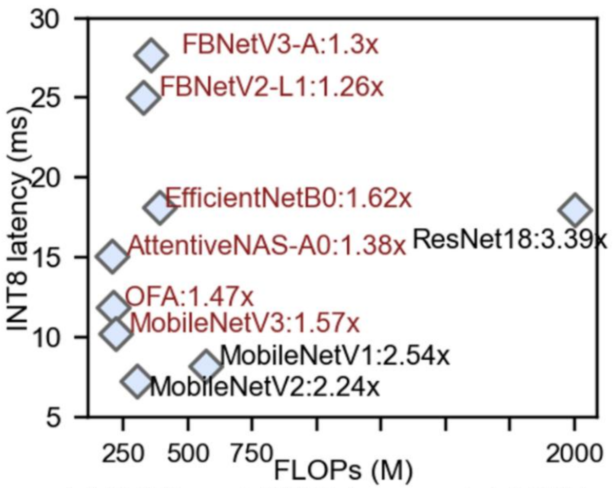


Speed up

Reduce the computations and the inference time

Introduction

- INT8 quantization only achieve only marginal speedup
- Traditional quantization methods focus on minimizing accuracy loss for a given pre-trained model, but ignore the real-world inference efficiency.



(a) FLOPs and INT8 latency on Intel CPU

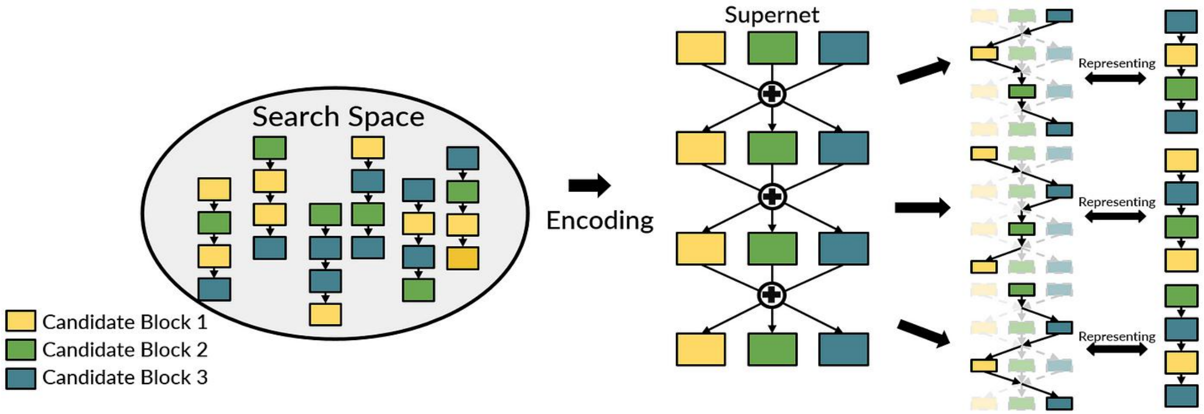
Less than 2x speedup

Still high latency !

Difficult to deploy on latency-critical scenarios

Introduction

- Neural Architecture Search (NAS) is a powerful tool for automating efficient quantized model design.
- Two-stage paradigm (one shot NAS)
  - step1: trains a weight-shared quantized supernet
  - step2: uses typical search algorithms to find subnets with best quantized accuracy under different FLOPs constraints



poor performance !

Reason: prior art search spaces cannot be well applied to quantization on diverse devices

Current search space limit NAS to find better quantized models for edge devices.

Introduction

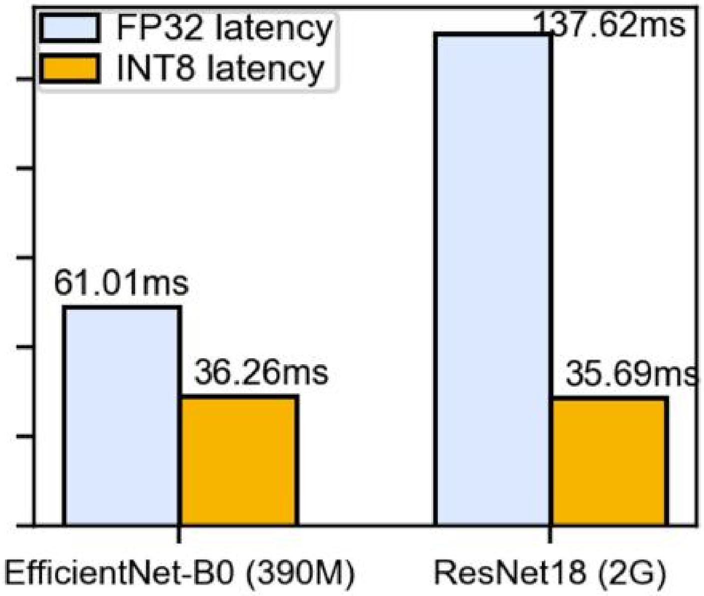
Question:

Can we design a hardware-friendly search space, allowing NAS to discover better models that meet the low INT8 latency and accuracy requirements ?

On-device Quantization Efficiency Analysis

What factors lead to current NAS search space quantization unfriendly issue?

Observation 1: FP32 latency and FLOPs are not good indicators of INT8 latency



(b) FP32 and INT8 latency on Pixel 4

A common belief is that model with lower FLOPs or FP32 latency leads to less INT8 latency

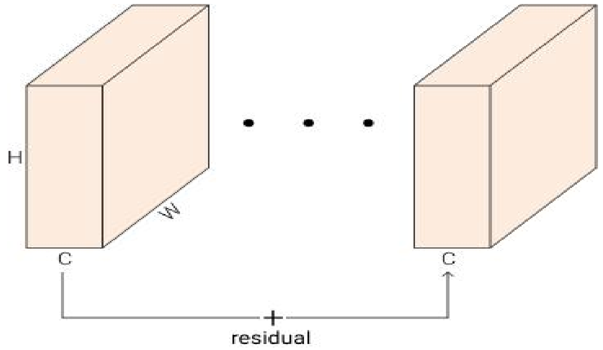
Neither of them is a good indicator of INT8 latency

On-device Quantization Efficiency Analysis

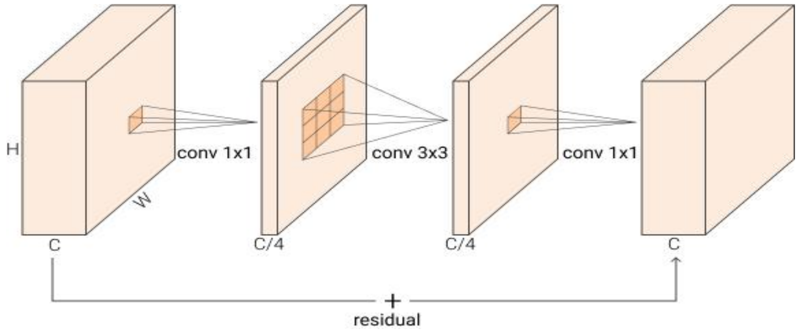
What factors lead to current NAS search space quantization unfriendly issue?

Observation 2: The choices of operators' types and configurations greatly impact the INT8 latency

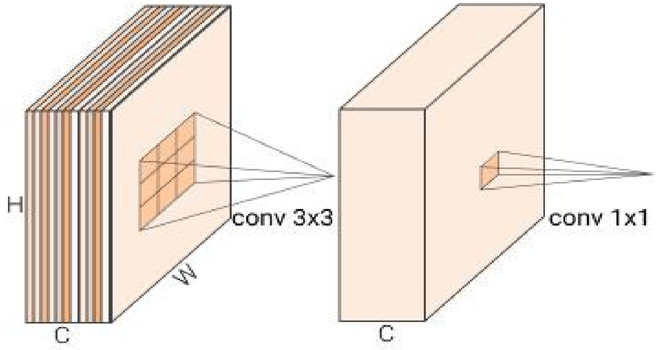
- The search space comprises a sequence of blocks.
- The block type varys and is allowed to search from a range of hyperparameter configurations
  - eg. kernel size, expansion ratio, channel width and depth…



residual



residual bottleneck



MBConv

On-device Quantization Efficiency Analysis

What factors lead to current NAS search space quantization unfriendly issue?

Observation 2: The choices of operators' types and configurations greatly impact the INT8 latency

Many block type and configuration choices in current search spaces unexpectedly ***slow down*** the INT8 latency !

Operator	Intel CPU	Pixel 4
Conv	2.6x	2.5x
DWConv	1.2x	2.0x
SE	0.7x	1.4x
Hardswish	0.7x	0.5x
Swish	0.8x	2.1x

(a) Avg. INT8 latency speedup (relative to FP32) of major operators in MobileNetV3.

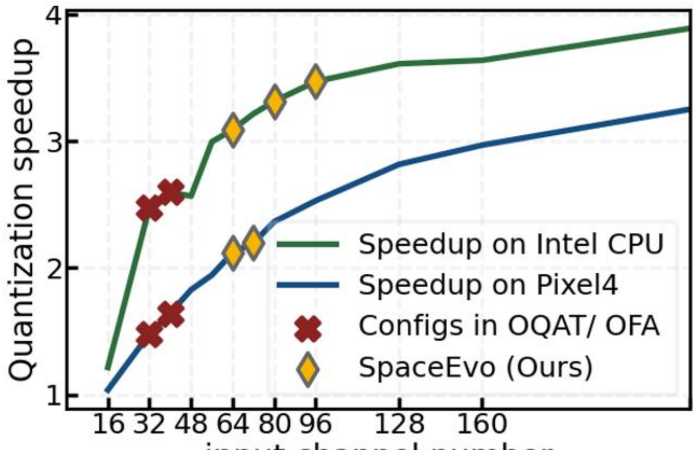
Some lightweight operators become slower because of the data transformation between INT32 and INT8

On-device Quantization Efficiency Analysis

What factors lead to current NAS search space quantization unfriendly issue?

Observation 2: The choices of operators' types and configurations greatly impact the INT8 latency

The configuration choices also determine the quantization efficiency



(b) Conv1x1 speedups

Small channel widths in OFA search space cannot benefit well from quantization

SpaceEvo can automatically design a search space with larger channel widths for better efficiency

On-device Quantization Efficiency Analysis

What factors lead to current NAS search space quantization unfriendly issue?

Observation 3: Quantization-friendly settings are diverse and contradictory across devices

The quantization-friendly operators are different and can be contradictory on diverse devices.

Operator	Intel CPU	Pixel 4
Conv	2.6x	2.5x
DWConv	1.2x	2.0x
SE	0.7x	1.4x
Hardswish	0.7x	0.5x
Swish	0.8x	2.1x

(a) Avg. INT8 latency speedup (relative to FP32) of major operators in MobileNetV3.

Quantization speedups are highly dependent on the inference engines and hardware

On-device Quantization Efficiency Analysis

What factors lead to current NAS search space quantization unfriendly issue?

Observation 1: FP32 latency and FLOPs are not good indicators of INT8 latency

Observation 2: The choices of operators' types and configurations greatly impact the INT8 latency

Observation 3: Quantization-friendly settings are diverse and contradictory across devices

There is no single structure (block types in a model) that is optimal for quantization on all hardware

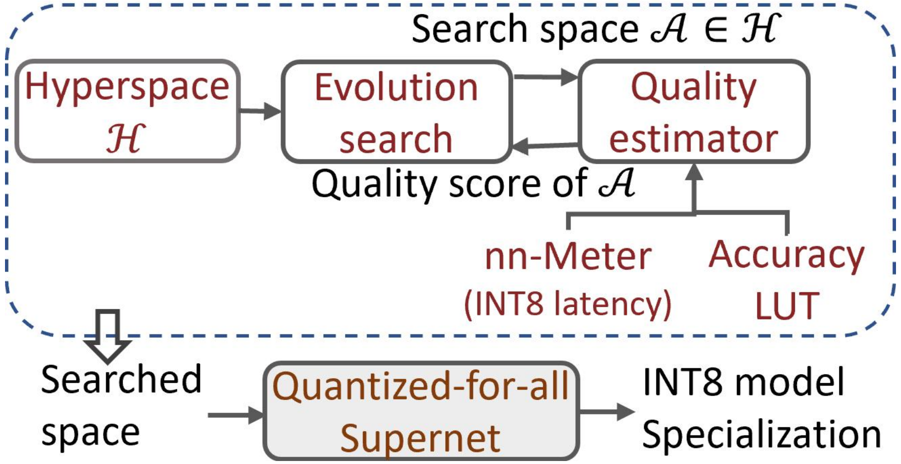
Design a ***specialized quantization-friendly search space*** for each hardware.

Each search space is tailored to the unique characteristics of the hardware and includes an optimal structure with elastic depths, widths, and kernel sizes.

Methodology

Core design concept

- Different from architecture search, where the goal is to **find the single best model from the space**
- we aim to find the **best search space** from lots of **all kinds of search spaces *(hyperspace)***
  - ***不同于神经网络搜索在一个搜索空间中找到最好的模型***
  - ***我们的目标是在一个不同搜索空间组成的更大的搜索空间（hyperspace）中，找到最好的一个搜索空间作为量化模型NAS的搜索空间。***



Methodology

Search Space Quality Score —— *如何评价hyperspace中的每一个search space的好坏?*

Latency-aware space quality of Q-T score.

Treat a space with good quality if its best searched subnets achieve optimal quantized accuracy under latency constraints T.

如果说一个search space 中最好的那个subnet在延迟限制下达到了最佳的量化精度，那么我们认为这样一个search space 是好的

用Q-T Score 来描述一个search space 的好坏

we treat every constraint equally important, and define Q-T score as the sum of each constraint: $\mathcal{Q}(\mathcal{A}, T_{1,\dots,n})=\mathcal{Q}(\mathcal{A}, T_1)+\mathcal{Q}(\mathcal{A}, T_2)+\dots, \mathcal{Q}(\mathcal{A}, T_n)$, $\mathcal{Q}(\mathcal{A}, T_i)$ is defined as:

$$\mathcal{Q}(\mathcal{A}, T_i) = \mathbb{E}_{\alpha \in \mathcal{A}, LAT(\alpha) \leq T_i}[Acc_{int8}(\alpha)] \qquad (1)$$

α是搜索空间中最佳的subnet

Acc_int8(α)是top-1 quantized accuracy evaluated on ImageNet validation set

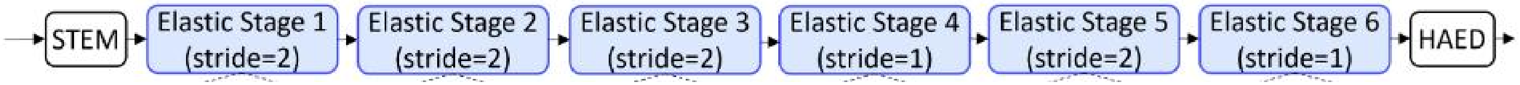
LAT(α)是量化后在目标设备上的延迟

Q-T Score就是在一定延迟限制下，获得最佳量化精度模型的数学期望，显然我们希望这个数学期望越大越好

Methodology

Elastic Stage and Problem Formulation

a chainstructured search space



Each stage defines a range of configurations c (e.g., kernel size, channel width, depth) for a specific block type b, and allows NAS to find the optimal architecture settings.

$$\mathcal{A} = STEM \circ E^{1}_{b,c}, \ldots \circ E^{N}_{b,c} \circ HEAD \qquad (2)$$

Methodology

Elastic Stage and Problem Formulation

Problem definition

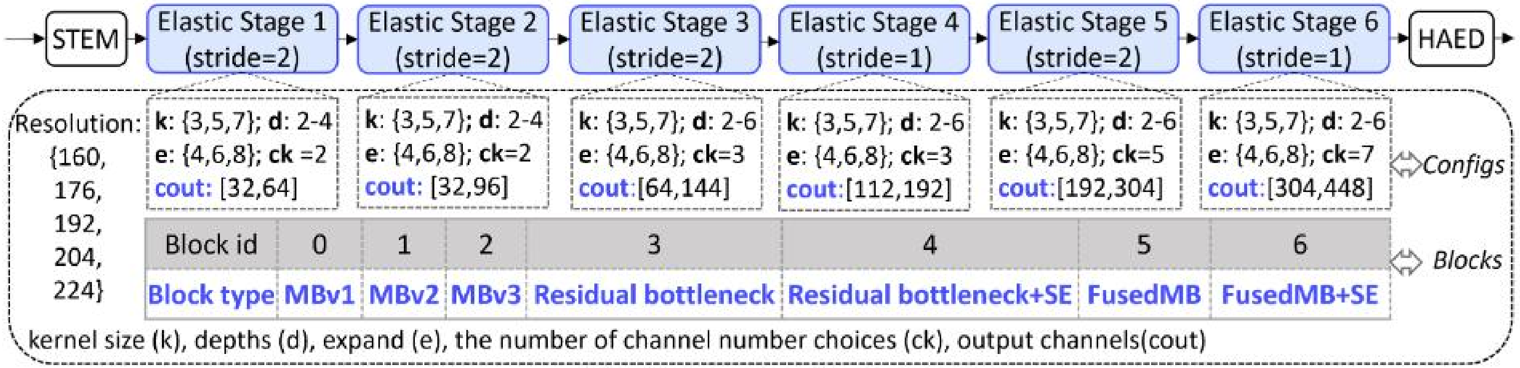
Operator type b and configuration c are two crucial objectives when searching quantization-friendly search space.
The task of space search can be simplified to find a search space with optimal elastic stages

$$\mathcal{A}(E^1_{b,c} \circ E^2_{b,c} \circ ... \circ E^N_{b,c})^* = \underset{E^i_{b,c} \in \mathcal{H}^i}{\arg\max}\, \mathcal{Q}(\mathcal{A}(E^1_{b,c} \circ E^2_{b,c} \circ ... \circ E^N_{b,c}), T)$$

Methodology

Searching the Search Space

Hyperspace design



(b) stage-wise hyperspace

A sampled search space is encoded by a sequential elastic stages.

Search by two dimensions

- Block type <u>*b*</u>
- Output channel width <u>*cout*</u>

Hyperspace has 10^9 candidate search spaces!

Methodology

Searching the Search Space

Evolutionary space search——aging evolution[1] *类似于遗传算法*

1. **我们首先随机初始化一组 P 个搜索空间，构成一个种群，其中每个采样空间被编码为** $(E_{b,c}^{1} \circ E_{b,c}^{2} \circ \ldots \circ E_{b,c}^{N})$
2. **对每一个采样的搜索空间都使用 Q-T score快速评估**
3. **接下来进入mutation iterations**
4. **在每次迭代中，我们再从hyperspace中随机采样S个搜索空间，并选择得分最高的候选者作为parent**
5. **mutation: 通过分别改变parent的block type 和 widths以生成两个child搜索空间。**
6. **我们评估这两个搜索空间的Q-T分数，并将它们添加到当前种群中。**
7. **最古老的 2个space 被删除，然后重复迭代。**
8. **在所有迭代完成后，我们收集所有采样空间并选择Q-T分数最高的空间作为最终搜索空间。**

[1](AAAI’19)Regularized Evolution for Image Classifier Architecture Search

Evaluation

Setup

- The INT8 latency constraints are {8, 10, 15, 20, 25} ms for Intel CPU and {15, 20, 25, 30, 35} ms for Pixel4.
- For each device, we search 5k search spaces in total and return the one with highest Q-T score.
- The population size P is 500 and sample size S is 125.

Evaluation

Comparison with SOTA search spaces.

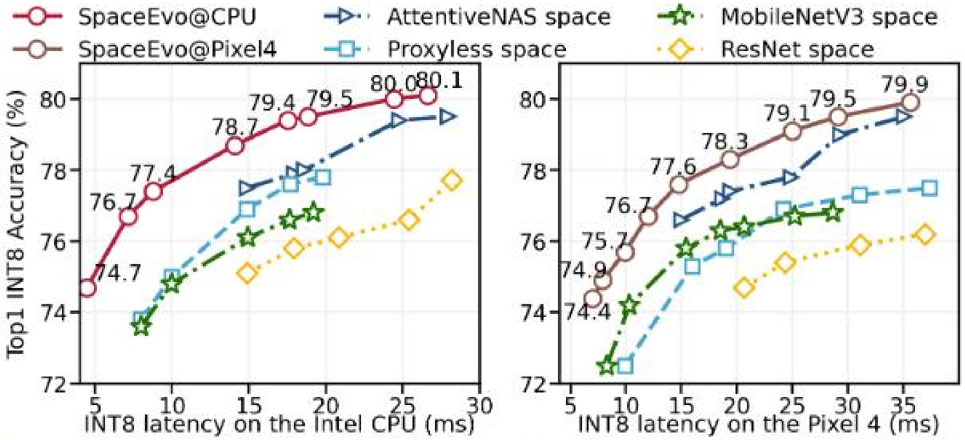


Figure 5. Best searched INT8 models with comparison to state-of-the-art NAS search spaces. Our searched spaces are proven to be the most quantization-friendly for the target device.

Evaluation

SpaceEvo under diverse latency constraints

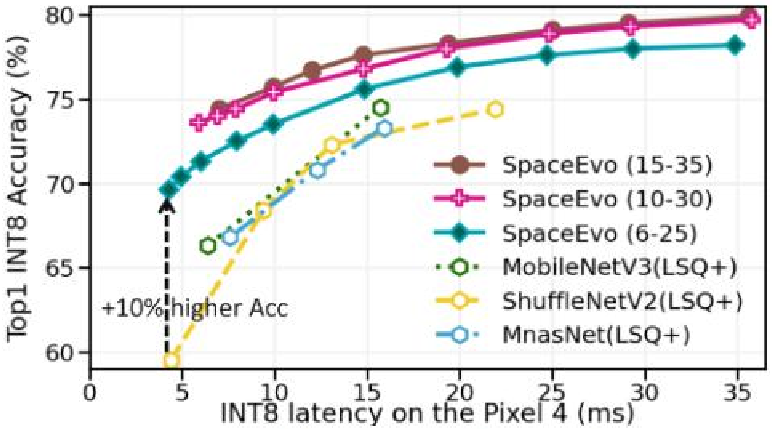


Figure 6. Search space design under diverse INT8 latency constraints. SpaceEvo (6-25 ms) delivers superior tiny INT8 models.

The Effectiveness of Discovered INT8 Models

Table 1. ImageNet results compared with SOTA quantized models on two devices. *: latency compared to FP32 inference.

(a) Results on the Intel VNNI CPU with onnxruntime

Model	Acc% INT8	CPU Latency INT8	CPU Latency speedup*	Acc% FP32	FLOPs
MobileNetV3Small	66.3	4.4 ms	1.1×	67.4	56M
SeqNet@cpu-A0	**74.7**	**4.4 ms**	**2.0**×	74.8	163M
MobileNetV2	71.4	7.3 ms	2.2×	72.0	300M
ProxylessNAS-R	74.6	8.8 ms	1.8×	74.6	320M
OQAT-8bit	74.8	9.8 ms	1.8×	75.2	214M
MobileNetV3Large	74.5	10.3 ms	1.5×	75.2	219M
OFA (#25)	75.6	11.2 ms	1.5×	76.4	230M
SeqNet@cpu-A1	**77.4**	**8.8 ms**	**2.4**×	77.5	358M
APQ-8bit	73.6	15.0 ms	1.5×	73.6	297M
AttentiveNAS-A0	76.1	15.1 ms	1.4×	77.3	203M
OQAT-8bit	76.3	14.9 ms	1.7×	76.7	316M
EfficientNet-B0	76.7	18.1 ms	1.6×	77.3	390M
SeqNet@cpu-A2	**78.5**	**14.1 ms**	**2.4**×	78.8	638M
APQ-8bit	74.9	20.0 ms	1.5×	75.0	393M
OQAT-8bit	76.9	19.5 ms	1.6×	77.3	405M
AttentiveNAS-A1	77.2	22.4 ms	1.4×	78.4	279M
AttentiveNAS-A2	77.5	22.5 ms	1.3×	78.8	317M
SeqNet@cpu-A3	**79.5**	**18.9 ms**	**2.6**×	79.6	981M
FBNetV2-L1	75.8	25.0 ms	1.2×	77.2	325M
FBNetV3-A	78.2	27.7 ms	1.3×	79.1	357M
SeqNet@cpu-A4	**80.0**	**24.4 ms**	**2.4**×	80.1	1267M

(b) Results on the Google Pixel 4 with TFLite

Model	Acc% INT8	Pixel4 Latency INT8	Pixel4 Latency speedup*	Acc% FP32	FLOPs
MobileNetV3Small	66.3	6.4 ms	1.3×	67.4	56M
SeqNet@pixel4-A0	**73.6**	**5.9 ms**	**2.1**×	73.7	107M
MobileNetV2	71.4	16.5 ms	1.9×	72.0	300M
ProxylessNAS-R	74.6	18.4 ms	1.8×	74.6	320M
MobileNetV3Large	74.5	15.7 ms	1.5×	75.2	219M
APQ-8bit	74.6	14.9 ms	2.0×	74.4	340M
OFA (#25)	75.6	14.8 ms	1.7×	76.4	230M
OQAT-8bit	75.8	15.2 ms	1.9×	76.2	287M
AttentiveNAS-A0	76.1	15.2 ms	2.0×	77.3	203M
SeqNet@pixel4-A1	**77.6**	**14.7 ms**	**2.2**×	77.7	274M
APQ-8bit	75.1	20.0 ms	1.9×	75.1	398M
OQAT-8bit	76.5	20.4 ms	1.8×	76.8	347M
AttentiveNAS-A1	77.2	21.1 ms	2.0×	78.4	279M
AttentiveNAS-A2	77.5	22.7 ms	2.0×	78.8	317M
SeqNet@pixel4-A2	**78.3**	**19.4 ms**	**2.3**×	78.4	402M
FBNetV2-L1	75.8	26.7 ms	1.5×	77.2	325M
OQAT-8bit	77.0	29.9 ms	1.7×	77.2	443M
FBNetV3-A	78.2	30.5 ms	1.5×	79.1	357M
SeqNet@pixel4-A3	**79.5**	**30.8 ms**	**2.1**×	79.5	591M
EfficientNet-B0	76.7	36.4 ms	1.7×	77.3	390M
SeqNet@pixel4-A4	**79.9**	**35.5 ms**	**2.2**×	80.0	738M

Thoughts

Strength

- automatically design a quantization-friendly space for target device, which delivers superior INT8 quantized models with SOTA efficiency on real-world edge devices.
- the proposed quality score (Q-T score) is a good approximate metric of the search space quality.
- Its potential extends to applications for other crucial deployment metrics, such as energy and memory consumption, enhancing the sustainability of edge computing solutions.

Weakness

- Experiments are mostly done with two mobile CPUs. It is uncertain whether the on-device quantization efficiency analysis in this paper still holds for mobile GPUs and even mobile NPUs or not.
- The Experiments devices are too few, which can not make the results or oberservation convincing.

Thank You !

Ye Wan 2023.11.20