Layer-Selective INT8 Quantization for MobileNetV2 via Depthwise-Focused QAT

Jay Katyan
University of Pennsylvania
Philadelphia, Pennsylvania, USA
jkatyan@seas.upenn.edu

Neel Shejwalkar
University of Pennsylvania
Philadelphia, Pennsylvania, USA
nshej@seas.upenn.edu

Norman Zhang
University of Pennsylvania
Philadelphia, Pennsylvania, USA
nzhang11@seas.upenn.edu

Abstract

Our project investigates the structural sensitivity of MobileNetV2 to INT8 quantization, motivated by the substantial accuracy degradation observed under standard post-training quantization (PTQ). Despite the memory and throughput advantages associated with 8-bit inference, we find that MobileNetV2's depthwise separable convolutions and narrow bottlenecks exhibit limited channel redundancy, making them highly vulnerable to quantization-induced feature drift. We performed a targeted sensitivity analysis to identify quantization-critical layers and evaluate a series of selective Quantization-Aware Training (QAT) strategies. Results show that partial retraining applied solely to classifier or depthwise layers is insufficient, whereas jointly retraining depthwise and 1x1 pointwise expansion layers yields significant accuracy. Finally, applying asymmetric activation quantization within this selective QAT framework closes the remaining accuracy gap, achieving 93.25% top-1 INT8 accuracy compared to 95.49% in FP32 and improving upon the performance of full QAT while retraining less than half of the model parameters. These findings demonstrate that full-model QAT is not necessary for MobileNetV2, and that structural layer selection combined with asymmetric quantization provides an effective pathway to high-accuracy, memory-efficient deployment.

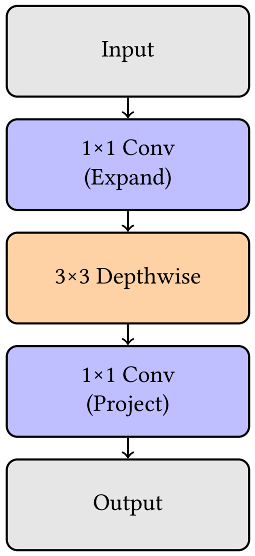


Figure 1: Simplified inverted residual block in MobileNetV2. The depthwise convolution (orange) operates on channels independently, making it highly sensitive to quantization noise.

1 Introduction

The deployment of deep learning models on edge devices is constrained by limited memory bandwidth and compute power. While standard INT8 quantization provides a viable solution by reducing precision from 32-bit floating point, it relies on the model's inherent robustness to noise. Standard convolutional networks exhibit such robustness through cross-channel summation, where quantization errors across several input channels are statistically averaged, resulting in natural noise suppression. In contrast, MobileNetV2 [4], a small and standard architecture for mobile vision tasks, presents unique challenges in quantization due to the model's "Inverted Residual" structure.

In order to achieve efficient inference on mobile devices, MobileNetV2 leverages inverted residual blocks with depthwise separable convolutions (see Figure 1). We provide a more detailed quantizable MobileNetV2 design shown in Figure 2. While this architecture enables real-time performance on resource-constrained devices, it fundamentally compromises the network's robustness to quantization. Unlike standard convolutional layers where quantization errors are averaged across hundreds of input channels, depthwise convolutions operate on single channels independently. As depthwise separable convolutions lack the channel redundancy required to absorb quantization error, noise introduced by rounding can lead to significant feature drift. This problem is compounded by MobileNetV2's usage of narrow bottleneck layers (with as few as 16 channels) combined with ReLU activations, causing minor quantization errors to degrade the limited feature information flowing through bottleneck projections, leading to catastrophic accuracy degradation.

Our preliminary results confirm the model's extreme sensitivity to quantization errors. Applying Post-Training Quantization (PTQ) with per-channel weight quantization to a fine-tuned FP32 MobileNetV2 model results in a drop from 95.49% top-1 accuracy on CIFAR-10 to 67.87% in the quantized INT8 model. The degradation is even more severe with per-tensor PTQ, resulting in an accuracy of just 44.18%. In contrast, full Quantization-Aware Training (QAT) recovers most of this lost accuracy by simulating quantization effects during training, achieving 92.98% accuracy. However, full QAT requires retraining all ~2.2M parameters in the network. This raises the question of whether full model retraining is necessary to preserve accuracy under quantization, or we can achieve comparable results by selectively retraining only the most quantization-sensitive layers.

In this work, we conduct a systematic sensitivity analysis to determine the minimal Quantization-Aware Training (QAT) scope required to match full QAT performance. We perform targeted

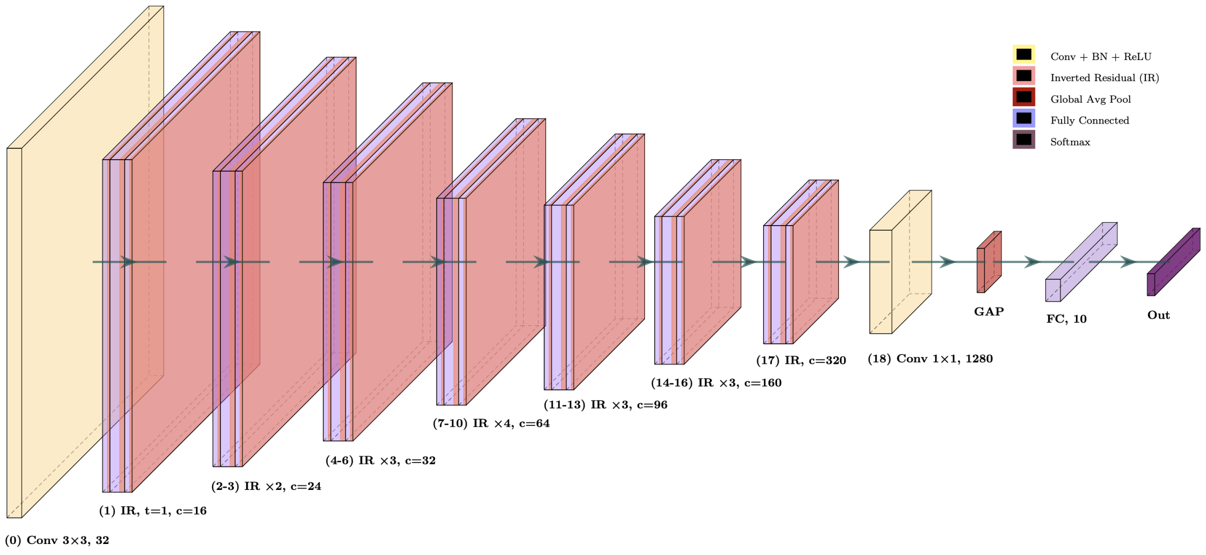


Figure 2: Quantizable MobileNetV2 architecture for CIFAR-10. The network processes 32x32 input images through an initial convolutional block (yellow), 19 inverted residual blocks (pink) organized in groups with increasing channel capacity, a final 1×1 convolution (yellow), global average pooling (red), and a fully-connected classifier (purple) outputting 10 classes. Each inverted residual block consists of 1×1 expansion, 3×3 depthwise convolution, and 1×1 projection layers.

layer-wise drift analysis to identify quantization-critical layers and evaluate multiple selective QAT strategies with varying retraining scopes. Our investigation tests the hypothesis that only structurally vulnerable layers—specifically depthwise convolutions and their adjacent 1x1 pointwise expansions—require retraining, while the remainder of the network can remain frozen. Our contributions are as follows:

(1) We demonstrate that MobileNetV2's depthwise separable convolutions are structurally hypersensitive to quantization noise, with per-tensor PTQ achieving 44.18% accuracy and per-channel PTQ reaching only 67.87%, compared to the 95.49% FP32 baseline. This confirms that standard PTQ approaches are insufficient for this architecture.
(2) Through systematic ablation studies, we show that partial retraining strategies targeting only the classifier (0.6% of parameters) or depthwise layers alone (3.4% of parameters) are insufficient. Classifier-only QAT achieves 87.67% accuracy, while depthwise-only QAT reaches 90.63%, both falling short of full QAT performance.
(3) We establish that selective QAT applied to depthwise and 1x1 pointwise expansion layers (43.6% of parameters) achieves 92.55% accuracy, nearly matching full QAT. When combined with asymmetric activation quantization to better utilize the UINT8 range for ReLU6 activations, this approach achieves 93.25% accuracy—an improvement compared to full QAT—while retraining less than half the model parameters and reducing model size from 9.18MB to 2.64MB.

2 Related Work

2.1 Quantization-Aware Training

Quantization-aware training (QAT) was first introduced by Jacob et al. [2], demonstrating that the addition of fake quantization operations during training enables models to adjust to the quantized value range while keeping the computation graph differentiable. This work showed how forward passes could be adapted to emulate INT8 inference while still using floating-point values to compute gradients, ultimately resulting in networks learning weights tolerant to quantization noise. Although this technique has become the standard for achieving low-precision deployments with minimal accuracy degradation, standard QAT suffers from its requirement to retrain the entire model with fake-quantization nodes. As such, the question of whether selective layer retraining can achieve comparable results at reduced training cost remains largely unexplored, particularly for architecturally constrained networks like MobileNetV2.

2.2 Depthwise Separable Convolutions and Quantization Sensitivity

MobileNetV2, introduced by Sandler et al. [4], uses inverted residual blocks with depthwise separable convolutions to significantly reduce computational complexity, but its enhanced architectural efficiency comes at the expense of increased sensitivity to quantization error. Yun and Wong [8] demonstrate that MobileNets experience major accuracy drops under PTQ (post-training quantization) because depthwise layers exhibit large channel-to-channel activation range differences, causing quantization errors to accumulate as information propagates through the network. Their analysis reveals low average precision per channel compared to each layer's

Table 1: Overview of QAT configurations used in our ablation study. DW = depthwise convolution, PW = pointwise (1x1) convolution.

Configuration	Layers Updated	Params Retrained	Quant Scheme
Full QAT	All layers	100%	Symmetric
Classifier-only	FC layer	0.6%	Symmetric
Depthwise-only	DW	3.4%	Symmetric
Depthwise + 1x1	DW + PW	46.5%	Symmetric
Depthwise + 1x1 (Asym)	DW + PW	46.5%	Asymmetric

full quantization range, with errors compounding across sequential depthwise operations.

Sheng et al. [5] also find that depthwise convolutions often include channel-specific outliers that consume critical quantization bits on zero-valued channels while under-representing informative channels. Unlike standard convolutional networks, where cross-channel summation naturally suppresses quantization noise, depthwise separable models lack the redundancy necessary to absorb quantization errors, making post-training quantization particularly unstable.

2.3 Selective and Mixed-Precision Quantization

While prior work establishes that full QAT can recover accuracy for MobileNetV2, the minimal scope required for retraining remains unclear. Some studies focus on mixed-precision schemes that assign different bit-widths to different layers based on sensitivity analysis [7], while others propose architectural modifications or specialized training procedures. However, existing approaches either apply full-network QAT or favor architectural modifications and mixed-precision schemes. In our study, we address this gap by systematically ablating QAT scope across MobileNetV2's layer types to determine whether depthwise layers alone, depthwise and pointwise layers together, or full network retraining is necessary. Our experiments further investigate whether asymmetric quantization schemes optimized for ReLU's non-negative output range can provide measurable improvements over symmetric approaches for this architecture, ultimately demonstrating that targeted layer selection can achieve comparable performance to full-network QAT while reducing training costs by more than half.

3 Methodology

Our three-phase experimental setup investigates the minimal retraining scope required to achieve high-accuracy INT8 quantization for MobileNetV2. First, we begin by training a full-precision (FP32) baseline model on CIFAR-10, which establishes our performance reference and provides activation distributions for quantization analysis. We then apply post-training quantization (PTQ) to investigate structural sensitivity to quantization noise, identifying architectural components most affected by reduced precision. Finally, we conduct a series of quantization-aware training (QAT) ablation studies to isolate which model layers require retraining under quantization constraints. Throughout all phases, we retain fixed dataset, architecture, optimizer, and training hyperparameters to ensure that variation in performance can be attributed solely to quantization method, rather than confounding procedural factors.

3.1 Baseline Model

The FP32 baseline is derived from a pretrained MobileNetV2 implementation, adapted for CIFAR-10 classification. MobileNetV2 consists of 19 inverted residual blocks using depthwise separable convolutions and ~2.2M parameters total. We replace the final linear layer to output 10 classes and fine-tune the model for 60 epochs using SGD with an initial learning rate of 0.01, momentum of 0.9, and weight decay of 5×10^{-4}. Additionally, a `MultiStepLR` scheduler reduced the learning rate at epochs 30 and 45 by a factor of 0.1. This FP32 baseline achieved 95.49% top-1 accuracy on the test set and serves as our baseline comparison for all subsequent experiments.

3.2 Post-Training Quantization

To characterize the structural sources of quantization error in MobileNetV2, we apply PTQ under two widely used quantization schemes: per-tensor and per-channel quantization. Per-tensor quantization applies a single global scale factor to all weights within each layer, while per-channel quantization computes independent scale factors for each output channel to better handle heterogeneous activation ranges in depthwise convolutions. Prior to quantization, we perform layer fusion (Conv + BatchNorm + ReLU) to minimize numerical discrepancies introduced by operator boundaries. We insert symmetric INT8 quantization for weights and asymmetric INT8 quantization for activations, following standard practice in PyTorch eager-mode quantization. To evaluate structural sensitivity, we compute per-block activation drift by measuring L_2 distance between FP32 and INT8 activations across the test distribution. Specifically, we use the defined equation below to calculate activation drift for each layer l:

$$\text{Drift}(l) = \frac{\|A_l^{\text{FP32}} - A_l^{\text{INT8}}\|_2}{\|A_l^{\text{FP32}}\|_2}$$

By using this equation, we can see which layer types contribute most to quantization instability and motivate selective QAT in subsequent experiments.

3.3 Quantization-Aware Training Ablation Studies

Following the PTQ sensitivity analysis, we design a series of QAT experiments that progressively increase the number of trainable parameters under quantization constraints. Each configuration introduces fake quantization nodes during training to simulate INT8 precision, allowing affected layers to adapt to a reduced dynamic range during optimization. Table 1 summarizes the layer scopes, parameter percentages, and quantization schemes for each QAT configuration examined in our research. Using the table as a reference framework allows for consistent comparison across experiments and clearly defines how each selective QAT strategy differs in retraining scope compared with the FP32 baseline.

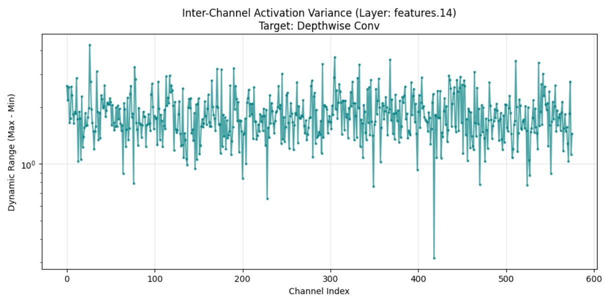


Figure 3: Dynamic ranges of activation distributions across all 576 channels of the depthwise convolution layer in bottleneck 14 (pre-quantization).

Experiment 1: Full-Network QAT: We begin with full-network QAT, in which fake quantization operations are inserted at all convolutional linear layers and all ~2.2M model parameters are updated. This establishes an upper bound on INT8 performance under optimal co-adaptation and serves as a reference for evaluating the effectiveness of restricted-scope QAT strategies.

Experiment 2: Classifier-Only QAT: To test whether quantization error can be corrected at the output stage alone, we enable QAT exclusively for the final fully connected classification layer while freezing all convolutional parameters. This experiment evaluates the hypothesis that feature distortion introduced in earlier layers may be recoverable by retraining only the classification head.

Experiment 3: Depthwise-Only QAT: Inspired by PTQ drift analysis indicating elevated sensitivity in depthwise convolutions, we selectively enable QAT for depthwise layers while freezing all pointwise (1x1) projection layers and classifier weights. This configuration tests whether retraining depthwise layers alone can compensate for per-channel activation variability inherent in depthwise operators.

Experiment 4: Depthwise + Pointwise 1x1 QAT: Given that depthwise and 1x1 projection layers operate as a coupled bottleneck structure, we extend selective QAT to both components simultaneously. Only parameters in depthwise and subsequent 1x1 expansion layers are retrained, while earlier feature extraction stages remain frozen. This configuration examines whether joint co-adaptation of depthwise and projection layers mitigates structural drift more effectively than depthwise retraining alone.

Experiment 5: Asymmetric Depthwise + Pointwise 1x1 QAT: Finally, motivated by activation distributions dominated by ReLU nonlinearities, we incorporate asymmetric activation quantization into depthwise + point-wise QAT configuration. In this experiment, weights and activations retain fake quantization nodes, but activation quantizers use per-tensor asymmetric scaling to leverage the full [0, 255] UINT8 range. This design choice evaluates whether asymmetric quantization schemes reduce saturation and clipping effects in intermediate bottleneck activations.

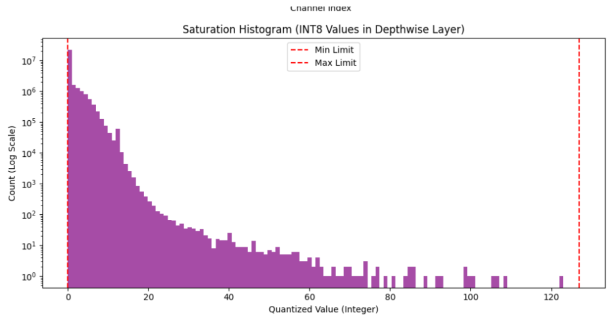


Figure 4: Post-quantized ranges of weights from the same layer as Figure 3.

4 Evaluation

We now present the empirical results of our quantization experiments and examine how different strategies affect model accuracy, computational cost, and structural robustness. The evaluation begins with post-training quantization (PTQ), followed by a progressive quantization-aware training (QAT) ablation study that isolate which architectural components must be retrained. All comparisons against the FP32 MobileNetV2 baseline, which achieves an accuracy of **95.49%** with a model size of **9.18MB**.

4.1 Post-Training Quantization

Applying the initial per-tensor PTQ scheme results in a highly reduced accuracy of **44.18%**, confirming that MobileNetV2 is highly sensitive to uniform scaling across channels. Analyzing the dynamic ranges in Figure 3 for the activations for a layer in the representative original FP32 model offers a clear reason why. We notice that the dynamic ranges of the activations vary wildly throughout the 576 channels of this particular layer. With per-tensor quantization calculating a single scale and zero-point across all channels, it is very feasible for many lower dynamic range channels to have their entire signal compressed to zero, resulting in the quantized network being unable to learn any meaningful representations. The saturation histogram in Figure 4 further confirms this mechanism. Most quantized values concentrate near the lower end of the UINT8 range, indicating that large activation values dominate the scale, while smaller activations are quantized with very coarse resolution or fully clipped into a single bin. This representational collapse produces the −27.62% accuracy gap shown in Table 2, demonstrating that per-tensor PTQ is unsuitable for MobileNetV2. This naturally motivated us to use per-channel PTQ with the *fbgemm* backend to mitigate these issues. We observe an improved, but still not passable, accuracy of **67.87%** with the per-channel scheme, confirming our prior expectations that naive quantization methods are not sufficient to achieve decent accuracy on this model. We notice an interesting trend in the relative error (drift) across the layers in Figure 6. Although the drift understandably shoots up as soon as the first bottleneck layer with sensitive 1x1 and depthwise convolutions is reached, rather than boundlessly increasing, it converges to around a value of 1.0 as the depth increases. This leads us to theorize that the network is operating in a different feature space than expected and has managed to learn an alternative representation

Table 2: Evaluation results for all quantization strategies. Accuracy Gap represents the difference between the FP32 baseline and the final INT8 accuracy. All models use identical architecture and training hyperparameters; only quantization strategy and retraining scope differ.

Method	Params Retrained	Model Size	Final INT8 Accuracy	Accuracy Gap
FP32 Baseline	–	9.18 MB	95.49%	–
PTQ (Per-Channel)	0%	2.64 MB	67.87%	−27.62%
Full QAT	100%	2.64 MB	92.98%	−2.51%
Classifier-Only QAT	0.6%	2.64 MB	87.67%	−7.82%
Depthwise-Only QAT	3.4%	2.64 MB	90.63%	−4.86%
Depthwise + 1×1 Down QAT	46.5%	2.64 MB	92.29%	−2.82%
Asymmetric DW + 1×1 Down QAT	46.5%	2.64 MB	93.25%	−2.46%

of the data, motivating us to design the methodology for our next set of experiments using QAT.

4.2 Quantization Aware Training (QAT)

Driven by the disappointing, yet expected, results from our PTQ experiments, we aim to examine the pareto-optimal boundary between training scope and accuracy to find the minimal number of parameters to retrain for high accuracy. Results are summarized in Table 2.

Experiment 1: Full-Network QAT (Reference): We establish a reference by first retraining all ~2.2M parameters for 10 epochs, resulting in a final accuracy of **92.98%**, only a 2.51% drop from the original fp32 baseline. This establishes the upper bound on quantized performance and confirms that MobileNetV2 can be quantized successfully if full co-adaptation is permitted.

Experiment 2: Classifier-Only QAT: For the second experiment, we retrain the 12k parameters associated with the final classifier, which amount to only 0.6% of the total ~2.2M parameters. Our initial suspicions from the final PTQ experiment are validated, as simply retraining the classification head results in the largest recovery in accuracy yet at **87.67%**, achieving much better results than any of the prior PTQ methods. This confirms that MobileNetV2's feature extractor is robust enough that quantization does not heavily affect the semantic content of the embeddings. However, the performance gap of approximately eight percentage points relative to full-network QAT indicates that quantization distortion introduced in earlier bottleneck blocks cannot be fully corrected by adjusting the classifier alone. Thus, classifier-only QAT provides a useful lower-bound reference but remains insufficient to restore deployment-level performance.

Experiment 3: Depthwise-Only QAT: Guided by the prior that the depthwise convolution layers in the inverted bottlenecks are generally sensitive to change and carry high information density per weight, our third experiment sought to add these parameters to our retraining scope (while retraining the classifier as well). While these only amount to $\approx$ 3.4% of the total parameters, we observe a demonstrable increase in accuracy to **90.63%**. As explained earlier, each channel in a depthwise convolution is quantized separately, so allowing these layers to adapt to the reduced precision restores their fine-grained feature extraction abilities.

Experiment 4: Depthwise + Pointwise 1x1 Down QAT: Depthwise convolutions and pointwise 1x1 convolutions operate as a tight structure and form the core of every inverted residual block, essentially defining the whole architecture. This next experiment adds the second, final pointwise 1x1 in each block to the retraining scope, raising the total percentage of retrained parameters to $\approx$ 46.5%. We observe a final INT8 accuracy result very close to the full QAT setup (**92.29%**) in Experiment 1, proving that once these are allowed to co-adapt during QAT, the rest of the architecture only provides marginal benefits.

Experiment 5: Asymmetric Depthwise + Pointwise 1x1 QAT: For our final experiment, we extend the depthwise + 1x1 selective QAT configuration by incorporating asymmetric quantization, mapping activation values to the full `[0,255]` UINT8 range. This choice increases representational resolution for non-negative activation distributions, which are inherently skewed toward zero due to ReLU nonlinearities (as evidenced in the saturation histogram in Figure 4). By aligning the quantizer dynamic range with the empirical activation statistics, asymmetric QAT reduces clipping and improves granularity in high-density regions of the distribution. We see this reflected in the results, achieving a final accuracy of **93.25%**, retaining the performance benefits of full QAT, but with better accuracy and fewer retrained parameters.

5 Discussion

In this work, we explored the structural sensitivity of MobileNetV2 to INT8 quantization, with a particular focus on identifying the minimal retraining scope required to maintain competitive accuracy. Through a systematic progression from post-training quantization to increasingly selective quantization-aware training, we demonstrated that accuracy degradation in quantized MobileNetV2 is neither uniform nor random: it exhibits predictably in depthwise separable bottleneck layers. The selective QAT experiments revealed that preserving model performance under quantization requires co-adaptation of depthwise and point-wise projection layers, while more superficial techniques such as retraining the classifier alone are insufficient.

5.1 Structural Insights from Selective Quantization

A key outcome of our study is the observation that quantization sensitivity in MobileNetV2 is structurally localized, not uniformly distributed across the network. Rather than affecting all layers equally, quantization noise gradually increases across inverted residual blocks and peaks in deeper bottleneck stages composed of depthwise separable convolutions paired with 1x1 projection layers. This pattern was evident from activation drift analysis, where later blocks exhibited higher drift values than early feature extraction layers (see Figure 5). The drift behavior directly motivated the layer scopes summarized in Table 1, where QAT configurations increasingly focused on depthwise and projection layers rather than uniformly retraining the entire model. From these structural insights, we can see that architectural design choices, particularly channel separation in depthwise convolution, play a central role in quantization robustness.

5.2 Practical Challenges in Selective QAT

Implementing selective QAT introduced several practical challenges that could shape how future quantization strategies should be designed. Freezing a large fraction of the model parameters required careful calibration of quantization node placement due to the fact that small shifts in fake quantization insertion or observer configuration affected training stability. The methodology summarized in Table 1 reveals that configurations with limited trainable parameters had greater sensitivity to quantizer settings, particularly depthwise-only QAT, where the drift persisted in deeper blocks despite retraining. Additionally, interpreting activation drift across sequential bottleneck layers was nontrivial. Drift accumulation could originate in depthwise layers while manifesting downstream in 1x1 projections. This complicated our initial assumption that sensitivity could be addressed by retraining depthwise layers alone, highlighting the need for layer grouping strategies rather than isolated layer updates. The challenges our group faced reflect broader issues in selective QAT research: retraining decisions must consider network topology, not just individual layer types.

5.3 Lessons for Deployment-Aware Model Compression

In general, our group learned two lessons from this research. First, quantization strategies benefit from diagnostic guidance rather than purely empirical or heuristic decisions. Simple metrics such as drift provided actionable information about where retraining resources should be invested, enabling us to refine QAT strategies as shown in Table 1. Starting from broad retraining, drift analysis helped isolate structural bottlenecks and progressively reduce the set of layers that required adaptation, ultimately arriving at a selective configuration that maintained INT8 stability without retraining the full network. Second, quantization performance is shaped not only by the number of parameters retrained but also by the interaction between layer structure and quantization scheme. In particular, activation distributions in depthwise and projection layers were highly skewed due to ReLU-based nonlinear compression, which made symmetric scaling insufficient. Incorporating asymmetric activation quantization allowed quantizers to better match local activation statistics and significantly reduce clipping in bottleneck layers, even when retraining scope remained limited. This observation suggests that effective quantization is a joint design problem.

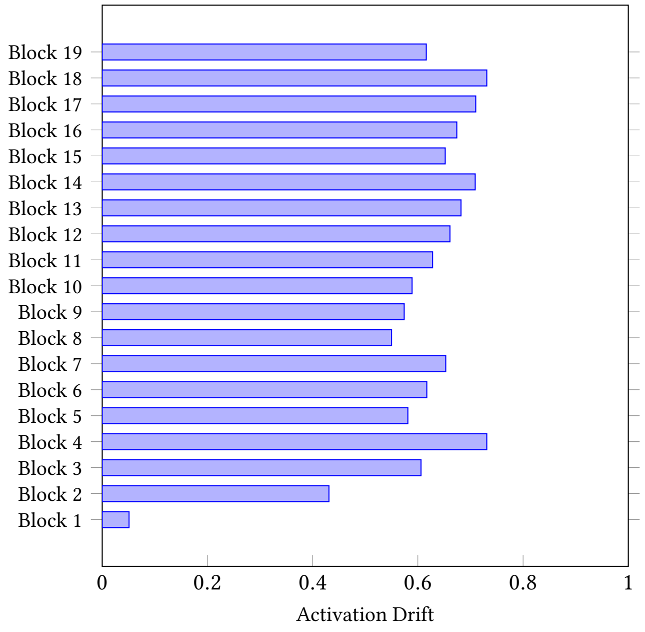


Figure 5: Activation drift across MobileNetV2 inverted residual blocks, computed as normalized L2 difference between FP32 and INT8 activations. Drift concentrates in deeper bottleneck structures featuring depthwise + 1×1 projection layers.

The big takeaway from our group is that layer selection and quantization scheme selection must be considered together, rather than treated as independent implementation choices.

6 Future Implementations

Although our investigation demonstrates the effectiveness of selective QAT with asymmetric activations for MobileNetV2, several questions remain regarding the generality and extensibility of this approach. First, our analysis focuses solely on the structural sensitivity of MobileNetV2 bottleneck layers, and it is unclear whether similar patterns appear in newer mobile architectures that incorporate squeeze-and-excitation (SE) units or lightweight attention. Second, layer selection in our experiments was guided by manual drift inspection, which is a more systematic, automated approach that could scale this strategy to larger model families. Finally, while our study considers 8-bit activation quantization, further work is needed to understand whether structural sensitivity persists under more aggressive mixed-precision constraints. In this section, we outline several directions that our group can research and expand upon this current work in the future.

6.1 Architecture Generalization for Modern Lightweight Models

Recent mobile architectures, such as MobileNetV3 [1] and EfficientNet-eLite [6], incorporate squeeze-and-excitation mechanisms, h-swish activations, and modified bottleneck topologies that are significantly different from MobileNetV2. These design changes can alter quantization-critical regions and reduce or redistribute sensitivity across layers. A logical next step in our current work is to reproduce activation drift measurements per block in MobileNetV3 to determine whether drift remains concentrated in the depthwise + 1x1 bottlenecks or whether SE attention mitigates or redistributes drift. A proposed implementation approach would be to replicate the drift computation pipeline, generate sensitivity heatmaps by block type, rank blocks by drift magnitude, and re-run selective QAT on the top-N blocks only. Finally, and finally measure the minimal retraining scope required to achieve or come close to full-QAT accuracy.

6.2 Automated Drift-Guided Layer Selection

In our research, we manually selected retraining scope using drift visualization, which is not scalable to other architectures or very deep networks. A second direction is to replace identification of quantization-critical blocks with a data-driven mechanism that automatically determines which layers should undergo QAT. Recent work on hardware-aware automated quantization has shown promise in learning quantization policies that adapt to both network structure and device constraints using reinforcement learning [7]. Adapting this strategy for selective QAT would involve learning a policy that decides, for each layer, whether to freeze, symmetrically quantize, or asymmetrically quantize based on local activation statistics or estimated drift magnitudes. Such a system could incorporate reward signals tied to final INT8 accuracy, retraining budget, or latency constraints, producing an optimized retraining map without human intervention. This would transform selective QAT from a manually tuned procedure into an automated optimization problem, expanding its applicability to a wider range of architectures and deployment environments.

6.3 Low-Bit and Mixed-Precision Quantization

Finally, exploring the interaction between selective QAT and more aggressive quantization strategies is a good direction for future research. While our work demonstrates the effectiveness of selective INT8 QAT, it remains unclear whether structural sensitivity persists at lower bit widths such as INT4, where quantization noise is increasingly higher. Recent post-training quantization methods that use adaptive or learned rounding heuristics have achieved promising accuracy, even under 4-bit constraints [3]. Combining these techniques with drift-guided layer selection (see Section 6.2) could allow certain blocks to operate at INT4 precision while reserving INT8 or asymmetric activation quantization for the most sensitive layers, yielding a mixed-precision deployment strategy that balances model size, latency, and accuracy. This line of work would provide insight into whether selective QAT generalizes under aggressive compression, and whether the concepts of drift localization and layer co-adaptation can be leveraged to maintain accuracy in the space of ultra-low-precision inference.

References

[1] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. 2019. Searching for MobileNetV3. arXiv:1905.02244 [cs.CV] https://arxiv.org/abs/1905.02244

[2] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. arXiv:1712.05877 [cs.LG] https://arxiv.org/abs/1712.05877

[3] Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. 2020. Up or Down? Adaptive Rounding for Post-Training Quantization. arXiv:2004.10568 [cs.LG] https://arxiv.org/abs/2004.10568

[4] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2019. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381 [cs.CV] https://arxiv.org/abs/1801.04381

[5] Tao Sheng, Chen Feng, Shaojie Zhuo, Xiaopeng Zhang, Liang Shen, and Mickey Aleksic. 2018. A Quantization-Friendly Separable Convolution for MobileNets. In *2018 1st Workshop on Energy Efficient Machine Learning and Cognitive Computing for Embedded Applications (EMC2)*. IEEE. https://doi.org/10.1109/emc2.2018.00011

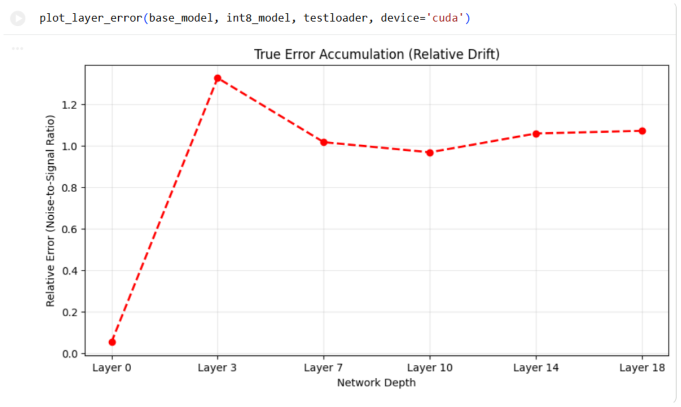
[6] Ching-Chen Wang, Ching-Te Chiu, and Jheng-Yi Chang. 2020. EfficientNet-eLite: Extremely Lightweight and Efficient CNN Models for Edge Devices by Network Candidate Search. arXiv:2009.07409 [cs.CV] https://arxiv.org/abs/2009.07409

[7] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. 2019. HAQ: Hardware-Aware Automated Quantization with Mixed Precision. arXiv:1811.08886 [cs.CV] https://arxiv.org/abs/1811.08886

[8] Stone Yun and Alexander Wong. 2021. Do All MobileNets Quantize Poorly? Gaining Insights into the Effect of Quantization on Depthwise Separable Convolutional Networks Through the Eyes of Multi-scale Distributional Dynamics. arXiv:2104.11849 [cs.CV] https://arxiv.org/abs/2104.11849

Appendix

A Extra Plots



```
plot_layer_error(base_model, int8_model, testloader, device='cuda')
```

Figure 6: Drift due to quantization across blocks.

B Technical Differences in MobileNetV2 Implementation

PyTorch's implementation of MobileNetV2 has several differences from the original paper [4] that are relevant to keep in mind for quantization performance and strategies.

(1) First, while the original architecture used ReLU6 ($\min(\max(0, x), 6)$) as the nonlinear activation function in all inverted residual blocks, PyTorch's *quantizable_mobilenet_v2* uses regular, unbounded ReLU, making the quantization problem significantly harder. This is important especially when it comes to examining PTQ failures. Unlike ReLU6, which has a bounded dynamic range (of 6), unbounded outlier activations in regular ReLU can heavily skew quantization statistics, leading to oversaturation in the lower end of the output range as shown in 4.

(2) Second, we note that the choice of downstream task strongly determines the relative importance of the classifier and how many parameters to retrain during QAT. While the original architecture outputted classification logits for ImageNet, the PyTorch implementation does so for CIFAR-10, resulting in the classifier size being reduced by 2 orders of magnitude from approximately 1280x1000 to 1280x10, and from 36.5% of the total model size to only 0.57%, a massive decrease. The ImageNet classifier simply absorbs much more of the representational capacity compared to the CIFAR-10 classifier. This is clear from experiment 2, where retraining the classifier led to good results but was ultimately too small to completely correct quantization drift in the backbone. Had we been working with the original version of the architecture, it is entirely possible that focusing the majority of the retraining efforts on the classifier would have yielded more fruitful compared to the backbone.