HySquant: Post Training Quantization Policy Search for Hybrid Model

Shaibal Saha*
shaibalsaha@oakland.edu
Oakland University
Rochester Hills, Michigan, USA

Yunge Li*
yungeli@oakland.edu
Oakland University
Rochester Hills, Michigan, USA

Abstract

Hybrid models demonstrate superiority in computer vision tasks by leveraging the local feature extraction of convolutional neural networks (CNNs) and the global dependency modeling of transformers. However, their heterogeneous characteristics make it difficult for a unified quantization policy to balance accuracy and efficiency. We propose a novel hybrid model post-training quantization policy search framework (HySquant), which is designed to automatically generate layer-by-layer mixed precision quantization policies that adapt to the CNN-Transformer hybrid architecture. The proposed framework is based on an optimal search engine that dynamically combines hardware constraints to explore the quantization design space and automatically allocates the optimal quantization bit width to each layer, thereby significantly improving hardware inference efficiency while ensuring model accuracy. Unlike existing quantization methods with fixed strategies or single architectures, this framework can flexibly adjust the quantization policy according to the characteristics of the input model based on the hardware requirements and output an optimized quantized version without retraining the model. Experimental results show that this method significantly improves hardware efficiency on hybrid models such as MobileViT while maintaining competitive accuracy, demonstrating clear advantages over existing methods. The proposed framework will provide a general and automated solution for efficiently deploying hybrid models, promoting the practical application of complex deep learning models in resource-constrained environments. We plan to release our code at https://github.com/shaibal13/HySqyant

CCS Concepts

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Keywords

Post training quantization, Search Quantization policy, Hardware efficiency, Hybrid model

*Both authors contributed equally to this research.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Shaibal Saha and Yunge Li. 2018. HySquant: Post Training Quantization Policy Search for Hybrid Model. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

With the widespread application of deep learning in the field of computer vision, the complexity and size of models continue to increase. Hybrid Models, which combine the architectural advantages of Convolutional Neural Networks (CNN) and Transformer, have become a key choice for improving task performance. CNN is good at extracting local features, while Transformer has significant advantages in global dependency modeling [7]. By combining the characteristics of both, hybrid models achieve state-of-the-art performance in tasks such as image classification, object detection, and semantic segmentation. However, the high computational requirements and storage overhead of these models severely restrict their deployment on resource-constrained devices such as mobile devices and edge platforms. Model compression technology is an important means to alleviate this problem. Quantization, as a technology that converts high-precision floating point parameters into low-precision integer representation, can significantly reduce the storage occupation and computational complexity of the model [6, 9]. Quantization can maintain high model accuracy while reducing model inference costs. Compared with Quantization-Aware Training (QAT), which requires retraining the model, Post-Training Quantization (PTQ) does not require retraining [1, 3]. It requires only a small amount of calibration data, making it an ideal choice for rapid model deployment. However, traditional PTQ methods mostly focus on a single architecture such as CNN or Transformer, and it is difficult to cope with the heterogeneous characteristics of hybrid models.

The complexity of hybrid models creates entirely new challenges for the design of PTQs. The local convolution characteristics of CNN are more robust to low-precision quantization, while modules such as multi-head attention and LayerNorm in Transformer are highly sensitive to numerical accuracy. Simple and unified quantitative strategies often fail to take into account the needs of both modules. In addition, the sensitivity of different layers in a hybrid model to quantization also varies significantly, with some key layers requiring higher precision to maintain performance, while other layers can accept lower precision to improve computational efficiency. Therefore, designing an automated and adaptive quantization strategy search framework to generate layer-by-layer mixed precision quantization strategies for hybrid models is a key issue that needs to be solved in the field of model compression.

To solve these above issues, we propose a hybrid model post-training quantization policy search framework (HySquant) with the

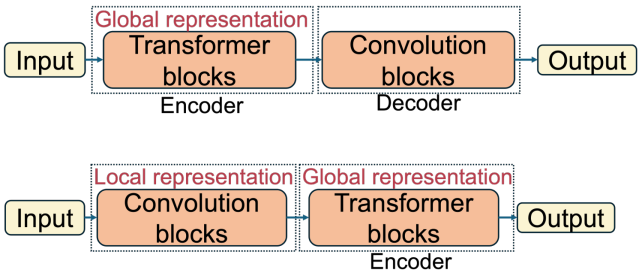

Figure 1: Hybrid architecture: two types hybrid model architectures

aim of providing an efficient automated quantization policy generation method for the hybrid architecture of CNN and Transformer. This framework is based on an optimal search engine and dynamically generates layer-by-layer quantization accuracy strategies by analyzing the structural characteristics and hardware constraints of the input model. For each input hybrid model, the framework can automatically output an optimized quantitative model, which significantly improves hardware inference efficiency while ensuring model accuracy. The framework does not require retraining the model, nor does it rely on manual debugging of quantization parameters, and is suitable for various hybrid architectures and hardware platforms.

The main contributions of this article include:

- We propose an automated post-training quantization strategy search framework designed specifically for hybrid models to solve the quantization optimization problem based on the hardware requirement.
- By combining an optimal search engine with a hardware feedback mechanism, a mixed-precision quantization strategy is dynamically generated, allowing the model to achieve optimal performance while meeting hardware resource constraints.

HySquant will provide a general and efficient solution for deploying hybrid models efficiently. It also opens up a new direction for applying post-training quantization technology in complex architectures.

2 Related work

PTQ is one of the major quantization techniques for neural networks without any training. By quantizing the model after training without retraining it, PTQ reduces computational complexity and storage requirements. Compared with QAT, PTQ has the advantages of low computational overhead and simple operation, which makes it more suitable for deployment on resource-constrained devices such as edge devices. However, existing PTQ methods are mainly oriented toward single architectures such as CNN or Transformer, and support for hybrid models is still insufficient.

In the field of PTQ, EasyQuant [11] is a basic but efficient quantization method that minimizes the cosine distance between the quantized output and the full-precision output by optimizing the scaling factors of weights and activations layer by layer. This method is suitable for traditional CNN models, but since the Transformer module is more sensitive to low-precision quantization in activation functions and attention mechanisms, EasyQuant is difficult to directly apply to hybrid architectures. To solve the quantization problem of the Transformer module, PTQ4ViT [12] proposed a Twin Uniform Quantization method to reduce the quantization error by special optimization of the GELU activation function and the Softmax operation. This method effectively improves the quantization performance of the Vision Transformer, but its design has not been extended to the hybrid architecture of CNN and Transformer.

HyQ [4] designed a hardware-friendly PTQ method for the CNN-Transformer hybrid architecture, which optimizes the unified quantization of CNN and Transformer modules by introducing the Quantization-Aware DeQuantization Strategy (QADS) and linear Softmax approximation. However, HyQ uses a fixed quantization strategy, which cannot adapt to the hierarchical characteristics of different models and lacks the ability to search for mixed precision strategies. In contrast, HAQ [9] proposes a framework for automatically generating quantization strategies through reinforcement learning. HAQ dynamically generates the optimal quantization bit width for each layer based on hardware feedback and model sensitivity. Although HAQ has achieved remarkable results in QAT and mixed precision quantization, it is mainly aimed at quantization methods that require training and fails to adapt to the needs of post-training quantization fully.

Although existing work has made progress in PTQ and automatic quantization policy generation, there are still significant challenges in quantization policy search in hybrid models. The module characteristics of CNN and Transformer are very different: CNN is more robust to low-precision quantization, while the transformer's multi-head attention, LayerNorm, and other modules have high accuracy requirements. This heterogeneity makes it difficult for a single strategy to balance the accuracy and efficiency of the hybrid architecture. In addition, most current methods rely on fixed quantization strategies and lack an automated, adaptive quantization framework for specific hybrid models.

To address the above problems, this paper proposes a PTQ policy search framework for a hybrid model, which focuses on post-training quantization policy search for hybrid architecture models. Similar to the reinforcement learning framework of HAQ, our framework automatically generates layer-by-layer mixed precision quantization strategies by analyzing the input CNN-Transformer hybrid architecture model and outputs an optimized quantization model. Compared with existing methods, our framework can not only dynamically adapt to heterogeneous modules in hybrid architectures, but also combine hardware characteristics and model requirements to improve hardware inference efficiency while maintaining model accuracy significantly. This study provides new ideas for hybrid architecture PTQ strategy search and fills the research gap in this field.

3 HySquant framework

We propose an efficient PTQ policy searching framework named HySquant for a hybrid model based on hardware requirements. This framework will allow users to automatically find the quantization policy for each layer given a pre-trained network to meet a platform's resource budget while maximizing accuracy. HySquant

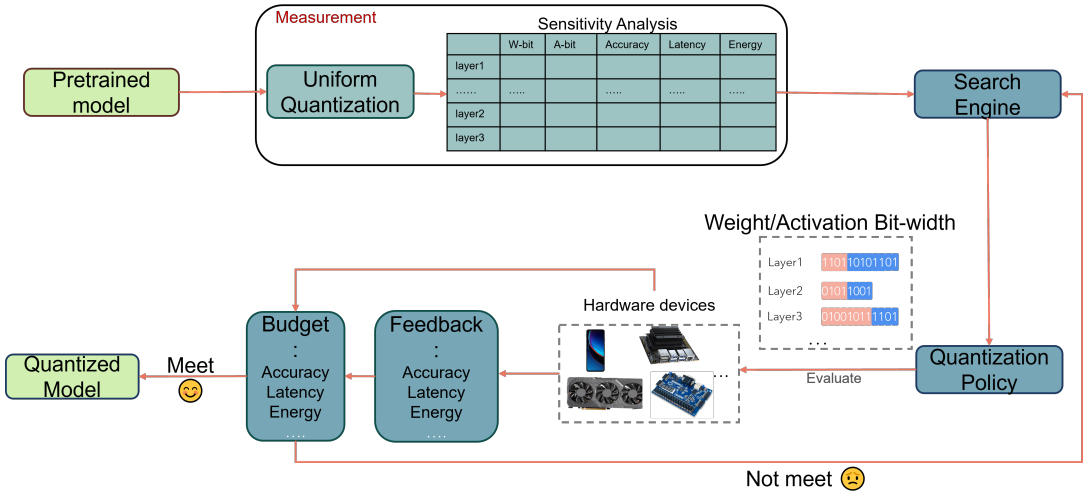

Figure 2: HySquant automatically adapts a pre-trained network given a resource budget for an edge device. This workflow is guided by resource consumption metrics. HySquant generates the post-training policy at each iteration to get the best-quantized hybrid model for the specific edge device.

is guided by sensitive analysis for resource consumption. The sensitive analysis is evaluated using efficiency measurements and the accuracy for each layer, thus eliminating the requirement for detailed platform-specific knowledge.

Figure 2 illustrates the overview of the HySquant framework. As we consider finding the optimized PTQ policy for specific hardware requirements, HySquant starts with the pre-trained hybrid model. Then, we evaluate each layer with different bit-widths in terms of top-1, top-5, latency, and energy consumption after applying quantization for individual layers. Next, we will propose a search design algorithm for efficient bit-width quantization techniques for each layer. Then, this suboptimal quantized model will be evaluated with the hardware budgets. This process will continue until HySquant does not find the optimal quantized model for the specific hardware. HySquant aims to solve the following constrained problem to find the budget-specific quantization policy for different edge devices.

$$\begin{aligned} &\text{maximize}_{acc} \quad Acc(pretrained) \\ &\text{subject to} \quad Quan_i(pretrained) \le Bud_i, \quad i = 1, \ldots, n, \end{aligned} \tag{1}$$

3.1 Uniform quantization

Although our main goal is to apply quantization techniques according to the layer requirements, we are currently only considering applying uniform quantization for each layer. We use a fixed scaling factor based on min-max calibration to apply uniform quantization. The scaling factor is derived from the layer-specific min-max statistics of weights within the range $[W_{\text{min}}, W_{\text{max}}]$. For n-bit uniform quantization, the quantized weight W_{quan} can be defined as follow:

$$W_{\text{quan}} = \left\lfloor \frac{\text{W} - W_{\text{min}}}{\Delta_W} + Z \right\rceil \tag{2}$$

$$\Delta = \frac{W_{\text{max}} - W_{\text{min}}}{2^n - 1}, n = 4, 8 \tag{3}$$

$$Z = \text{round}\left(\frac{-W_{\text{min}}}{\Delta}\right) \tag{4}$$

Here, we use n=4,8 for now. In the future, we plan to make it more robust from [2-8] bits, inspired by HAQ [9]. W_{quan} denotes the quantization function for each layer in the model. The input $W \in \mathbb{R}^{c \times h \times w}$ denotes the input with FP32 precision, where c is the number of channels and $h \times w$ is the height and width of the input. Moreover, W_{quan} denotes the n-bit integer value. The scaling factor, Δ, is calculated by mapping the range of the original tensor values between W_{max} and W_{min}. The zero point (Z) shifts the quantization so that the minimum value in the tensor is aligned with the quantization range. The range of the target quantization is calculated by $[0, 2^n - 1]$ and clamping it to make sure values are within the range. The values are rounded and clamping using the following function:

$$W_{\text{quan}} = \text{round}(\text{clamp}(W_{\text{quan}})) \tag{5}$$

Once the model has processed the input of the quantized blocks, we need to dequantize the values during inference. We do not need to change the weights/activations for these steps permanently. So, we dequantize the value on the fly to evaluate the quantization error and accuracy. The dequantize function (W_{dequan}) can be write as follows:

$$W_{\text{dequan}} = (W_{\text{quan}} - Z) \times \Delta \tag{6}$$

3.2 Sensitivity analysis

After applying uniform quantization in each layer, we evaluate this model as a sensitivity analysis and make the lookup table for further use. Figure 3 illustrates the generation workflow of the sensitivity analysis lookup table. Our lookup table comprises top-1, top-5, latency, and energy parameters. **Accuracy** In this sensitivity analysis, top-1 and top-5 are utilized as accuracy metrics as we currently focus on image classification. The top-1 and top-5 accuracy can be calculated as follow:

$$\text{Top-1} = \frac{1}{N} \sum_{i=1}^{N} 1\{\hat{y}_i = y_i\} \tag{7}$$

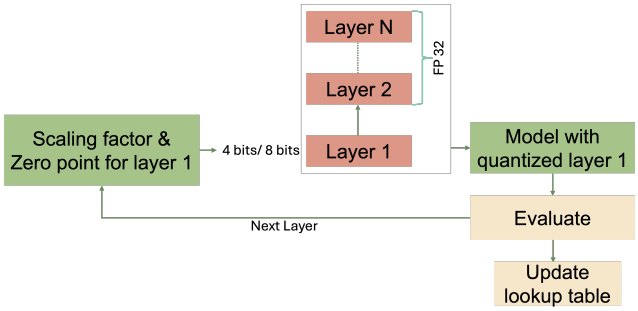

Figure 3: Workflow for building the sensitivity analysis lookup table

$$\text{Top-5} = \frac{1}{N}\sum_{i=1}^{N} 1\{y_i \in \hat{Y}_i\} \tag{8}$$

Here, in equation 7 and 8, N denotes total number of samples. $\hat{y}_i$ denotes Predicted class for the i-th sample (class with the highest probability) for top-1 accuracy. However, $\hat{y}_i$ in top-5 is the set of the top 5 predicted classes (ranked by probability) for the i-th sample. y_i denotes the ground truth label for the i-th sample in both equation. 1 is the indicator function that equals 1 if the condition $(\hat{y}_i = y_i)$ or $(y_i \in \hat{Y}_i)$ is true in the respective equation, and 0 otherwise.

Energy We utilize the number of flops from a specific hardware description for esteemed energy calculation. We can get the maximum power from an edge device. Power in any device, especially GPU, is calculated as equation 9. We only esteem the maximum power (m_p) by summing the p_{base} and p_{flops}, as p_{memory} is not needed in this current energy consumption calculation. Additionally, we can achieve the maximum number of flops (m_f) from the hardware description. We use these two hardware descriptions to calculate the power usage per flop (p_f) and multiply it with latency to calculate the energy consumption (E).

$$power = p_{base} + p_{flops} + p_{memory} \tag{9}$$

$$\text{Power per FLOPS } (p_\text{f}) = \frac{m_\text{p}}{m_\text{f}} \tag{10}$$

m_p : Maximum power of the device (in watts)

m_f : Maximum number of FLOPS

$$\text{Energy}(E) = p_\text{f} \times latency \tag{11}$$

This sensitivity analysis lookup table will later be used to find the quantized model through a search algorithm.

3.3 Hypothetical Workflow

The search engine will find the optimized quantization bit-widths and techniques based on the information gathered in the sensitivity analysis lookup table. The overall quantization policy will then be evaluated on hardware devices. Each hardware device has its budget in terms of accuracy, latency, and energy. The budget and the feedback from hardware edges have two possible scenarios: First, if the budget meets the feedback, the optimized quantized model will be output. Second, if the criteria are not met, then the framework will return to the search engine to find another quantization policy that meets the budget of that particular hardware device.

4 Experiment

In this section, we evaluate the sensitivity analysis of HyquanS. We use MobileViT_s [5] variants as our pre-trained model achieved from Timm library [10]. We evaluated our framework on ImageNet-1K [2] validation datasets. We evaluated all our experiments on the Nvidia RTX3080. The maximum power RTX3080 can achieve $320W$. We do not consider p_{memory}, about $80W$. So, the maximum power that flops can use is $240W$. From the hardware description, the maximum flops for this GPU are 28.77×10^{12} [8]. Figure 4 illustrates the sensitivity analysis lookup table on the Nvidia RTX3080 GPU.

Some rules can be found by analyzing the impact of different quantization policies on the overall performance and efficiency of the model for each layer. For example, as shown in the results in Figure 4, the $'stage.0.0.conv1\_1x1.conv'$ layer will greatly affect the model accuracy under the W4A4 and W8A4 quantization strategies, so we will consider only implementing INT8 quantization on the activation of this layer. By finding such rules, we will greatly reduce our search space and thus improve the efficiency of the entire search framework.

5 Conclusion

This paper proposes a novel hybrid model post-training quantization strategy search framework (HyQuanS), which aims to automatically generate layer-by-layer quantization strategies that adapt to the CNN-Transformer hybrid models, solving the problem of automatically generating hybrid model post-training quantization strategies. This framework combines the heterogeneous characteristics of the model and hardware constraints to dynamically allocate quantization bit width for each layer to achieve the best balance between model accuracy and hardware performance. Experimental results show that the framework achieves significant hardware efficiency improvements on hybrid models such as MobileViT, while maintaining competitive model accuracy. The research in this paper provides an efficient and versatile solution for the automatic generation of hybrid model quantification strategies, and opens up a new direction for the deployment of deep learning models in resource-constrained environments.

6 Future plan

At present, we have completed the overall design of the hybrid model post-training quantization policy search framework and clarified the basic structure and goals of the framework. In addition, we have completed the quantization sensitivity analysis of each layer of the hybrid model, laying a solid theoretical foundation for subsequent policy generation. Next, we plan to build a search engine or search algorithm to achieve automatic generation of layer-by-layer quantization policies while fully considering hardware constraints and model characteristics. We will also use MobileViT as the main test model to verify the actual performance of the framework on edge devices and expand it to other hardware platforms to evaluate the versatility. Finally, we will optimize the framework based on the experimental results, complete the paper writing, and look

	Layer	Weight Bitwidth	Activation Bitwidth	Top-1 Accuracy	Top-5	Latency	FLOPs	Energy
2	stem.conv	4	4	67.4	87.8	2.341743230819702	7077888	0.00013362119969768222
3	stem.conv	4	8	66.0	86.6	2.268092632293701	7077888	0.00012941865468592544
4	stem.conv	8	4	82.8	96.2	2.3649723529815674	7077888	0.0001349466666106819
5	stem.conv	8	8	85.6	97.2	2.2423219680786133	7077888	0.00012794816593886463
14	stages.0.0.conv1_1x1.conv	4	4	22.0	38.6	2.387953519821167	16777216	0.00032298189049378566
15	stages.0.0.conv1_1x1.conv	4	8	70.4	91.4	2.2680726051330566	16777216	0.0003067674356734968
16	stages.0.0.conv1_1x1.conv	8	4	37.8	59.4	2.254054546356201	16777216	0.0003048714276116896
17	stages.0.0.conv1_1x1.conv	8	8	85.0	97.0	2.2529807090759277	16777216	0.0003047261860933826

Figure 4: An example of how sensitivity analysis lookup table on Nvidia RTX3080 GPU store

forward to contributing this research to the field of quantization policy generation.

References

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

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[3] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*. Chapman and Hall/CRC, 291–326.

[4] Nam Joon Kim, Jongho Lee, and Hyun Kim. 2024. HyQ: Hardware-Friendly Post-Training Quantization for CNN-Transformer Hybrid Networks. In *IJCAI*.

[5] Sachin Mehta and Mohammad Rastegari. 2021. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178* (2021).

[6] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. 2019. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1325–1334.

[7] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. 2021. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems* 34 (2021), 23296–23308.

[8] NVIDIA. [n. d.]. GeForce RTX 30 Series Graphics Cards: RTX 3080 and RTX 3080 Ti. https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3080-3080ti/. Accessed: 2024-12-12.

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

[10] Ross Wightman. 2019. PyTorch Image Models. https://github.com/rwightman/pytorch-image-models. https://doi.org/10.5281/zenodo.4414861

[11] Di Wu, Qi Tang, Yongle Zhao, Ming Zhang, Ying Fu, and Debing Zhang. 2020. Easyquant: Post-training quantization via scale optimization. *arXiv preprint arXiv:2006.16669* (2020).

[12] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. 2022. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European conference on computer vision*. Springer, 191–207.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009