2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

GSE-MN4: Group-Shared Exponents Integer Quantization for MobileNetV4

Shenglin Yang[1[0000-0009-6533-1736]], Zhuo Han[2[0009−0001−3986−3167]], Hahn Yuan[3[0000−0001−7846−0240]], and Yanmei Hu[4[0000-0003-2295-1328]]

[1] Chengdu University of Technology, Chengdu, China
[2] Dalian University of Technology, Dalian, China
[3] Peking University, Beijing, China
[4] Chengdu University of Technology, Chengdu, China
yzh@pku.edu.cn

Abstract. This paper introduces Group-Shared Exponents Integer Quantization for MobileNetV4, a novel quantization framework tailored for efficient deployment of deep learning models on resource-constrained edge devices. Our method employs the Group-Shared Exponents (GSE) format, which shares exponents among groups of parameters and quantizes mantissas under a shared exponent constraint, significantly reducing memory overhead compared to traditional quantization techniques. Furthermore, we introduce an automated mixed-precision quantization scheme that allocates bit-widths based on layer sensitivity, thereby assigning each layer an optimal quantization bit-width. This strategy effectively optimizes the trade-off between accuracy and efficiency. Extensive experiments on the ImageNet1K dataset demonstrate that GSE-MN4 outperforms conventional quantization methods. For instance, the GSE-MIX quantization method on MN4-Conv-S achieves a Top-1 accuracy of 73.34\% with a memory footprint of only 3.23 MB, maintaining high accuracy while substantially reducing memory usage. Our work highlights the potential of GSE-INT for efficient and accurate deployment of deep learning models in mobile and edge scenarios.

Keywords: Post-Training Quantization · MobileNetV4.

1 Introduction

In recent years, the demand for Edge AI applications has grown significantly [26], with mobile networks, particularly edge networks [22], becoming a critical component of the modern computing ecosystem. This trend has driven the need for deploying deep learning models on mobile and IoT devices [6], spanning a wide range of applications such as smart homes, autonomous driving, and augmented reality [14]. Consequently, the development of lightweight and efficient neural networks that can be deployed on mobile devices has become a major research focus [23]. For resource-constrained edge devices, the compression of model size is particularly challenging [13]. For instance, some microcontrollers (MCUs) are equipped with only 16MB of memory, imposing stringent requirements on the memory footprint of neural networks.

MobileNetV4 (MN4) [19] has emerged as a promising solution for mobile and edge computing, striking a balance between model accuracy and computational efficiency. Although MN4 is inherently designed to be lightweight and efficient, further reduction in model size is necessary for deployment on devices with extremely limited memory, computational, and energy resources [8]. In such scenarios, model quantization has become a crucial technique. Quantization reduces the bit-width of model weights, significantly decreasing memory usage [10] and offering substantial advantages for deployment on edge devices. However, we observe that traditional quantization methods often lead to a notable degradation in performance when applied to MobileNetV4. This issue is particularly pronounced in smaller models with highly discrete weight distributions, such as MN4, where quantization increases numerical variance and introduces outliers [12]. These outliers significantly exacerbate quantization errors when traditional quantization techniques are employed [17].

Group quantization has been proposed as a refined approach [11] to address the issue of imbalanced weight distributions. However, this method incurs additional memory overhead due to the need to store extra FP16 scale and zero-point values for each group. Specifically, each group requires an additional 32 bits of storage for these parameters. Furthermore, the dequantization process at runtime introduces additional multiplication operations, further straining the already limited resources of edge devices.

To address these challenges, we propose **GSE-MN4: Group-Shared Exponents Integer Quantization for MobileNetV4**, a novel quantization framework that leverages the Group-Shared Exponents (GSE) integer format [30]. The GSE-INT method is specifically designed to mitigate the problem of large numerical variance in quantized models by employing a grouping mechanism. Simultaneously, it replaces the FP16 scale and zero-point with a 5-bit exponent, eliminating costly multiplication operations during dequantization. Additionally, we observe that different structures within MobileNet exhibit varying sensitivities to GSE-INT quantization. For example, convolutional layers with larger feature maps tend to be more sensitive to quantization errors compared to depthwise separable convolutions, which are more robust due to their reduced parameter count. To address this, we propose an automated mixed-precision quantization scheme [24]. Specifically, we first evaluate the sensitivity of each layer and then apply higher-bit GSE-INT quantization to more sensitive layers and lower-bit GSE-INT quantization to less sensitive layers. This approach achieves superior quantization results.

We validate the effectiveness of the GSE-INT method through extensive experiments on the MobileNetV4 architecture. Our results demonstrate that GSE-INT quantized models achieve significantly higher accuracy compared to traditional quantization methods under the same memory constraints. For instance, at 8-bit quantization, the GSE-INT quantized MN4-Conv-S model achieves a Top-1 accuracy of 72.6\%, with a memory footprint of only 2.7 MB (Original BF16 is 7.2MB). These results highlight the potential of GSE-INT in enabling efficient deployment of deep learning models on resource-constrained devices without compromising performance.

Our key contributions are as follows:

1. Group-Shared Exponents Integer Quantization for MN4. We introduce a quantization strategy that shares exponents among groups of parameters, enabling efficient

representation of model weights in integer format. This approach reduces the memory footprint of MN4, making it more suitable for resource-constrained mobile devices. (This is the first work to introduce GSE-Tuning into MobileNetV4 [28], enabling efficient deployment in mobile scenarios.)

2. Mixed-Precision Quantization. We conduct the first sensitivity analysis of MobileNetV4 to quantization and propose an automated mixed-precision quantization scheme that optimizes bit-width allocation based on layer sensitivity.

3. Pareto-Optimal Performance. Through extensive experimentation, we demonstrate that GSE-MN4 achieves a favorable trade-off between accuracy and efficiency, making it a practical solution for deploying MN4 on a wide range of mobile devices.

2 Related Work

2.1 Efficient Neural Network Architectures

The evolution of MobileNets has been marked by a continuous effort to balance accuracy and efficiency. MobileNet [8] introduced depthwise separable convolutions, significantly reducing the computational cost compared to traditional convolutions. MobileNetV2 [20] further improved efficiency by introducing inverted residual blocks with linear bottlenecks, which reduce the number of parameters while maintaining model capacity. MobileNetV3 [7] leveraged neural architecture search (NAS) to optimize the network structure for mobile devices, achieving state-of-the-art performance on various benchmarks. MobileNetV4 [19] represents the latest iteration in this lineage, introducing the Universal Inverted Bottleneck (UIB) block and Mobile Multi-Query Attention (MQA) to achieve Pareto-optimal performance across diverse hardware platforms. The UIB block unifies several prominent micro-architectures, including Inverted Bottleneck, ConvNext, and Feed Forward Network (FFN), offering flexibility in spatial and channel mixing. Mobile MQA, on the other hand, optimizes attention mechanisms for mobile accelerators, delivering significant speedups without compromising accuracy.

2.2 Quantization Techniques

Quantization has emerged as a critical technique for reducing the memory and computational demands of neural networks [6]. Traditional quantization methods, such as 8-bit integer (INT8) quantization [18], have been widely adopted due to their simplicity and compatibility with hardware accelerators. However, these methods often struggle to maintain accuracy in models with large dynamic ranges, such as MN4.

Recent advancements in quantization have focused on more sophisticated approaches to preserve model accuracy. Quantization-Aware Training (QAT) involves training the model with quantized weights and activations, allowing it to adapt to the reduced precision [1,21]. Post-Training Quantization (PTQ), on the other hand, quantizes a pre-trained model without additional training, making it more practical for deployment [25,29,27]. In this paper, we focus on the PTQ for MobileNetV4. Floating-point (FP)

quantization [16], particularly FP8, has gained attention for its ability to represent a wide range of values with reduced precision. However, FP8 quantization still incurs higher memory and computational costs compared to integer-based methods. This has led to the development of hybrid quantization techniques that combine the benefits of FP and INT quantization.

3 Method

3.1 Preliminary

Quantization is a widely adopted technique [18,4] in deep learning to reduce the computational complexity and memory footprint of neural networks by converting continuous floating-point values (e.g. weights, activations, and gradients) into discrete integer representations.

For a given floating-point tensor x, the standard b-bit INT quantization process can be formalized as follows:

$$x_{int} = Q(x) = clamp\left(\left\lfloor\frac{x-z}{s}\right\rceil, q_{min}, q_{max}\right) \tag{1}$$

where s is the scaling factor, which maps the range of floating-point values to the desired integer range; z is the zero-point, an offset that ensures proper alignment between the quantized and floating-point ranges; $\lfloor\cdot\rceil$ denotes the rounding operation (typically rounding to the nearest integer); clamp(·) restricts values to the valid integer range $[q_{min}, q_{max}]$, where $q_{min} = -2^{b-1}$ and $q_{max} = 2^{b-1} - 1$ for a signed b-bit representation.

Quantization significantly reduces the memory requirements [15] of neural networks, making them more suitable for deployment on resource-constrained devices.

Group Quantization extends traditional quantization by partitioning the tensor into multiple groups, each sharing a common scaling factor s_g and zero-point z_g [9]. This approach reduces the number of quantization parameters compared to per-channel quantization, thereby further optimizing memory usage and computational efficiency while maintaining competitive accuracy.

For a tensor x divided into G groups, the quantization process for group g is defined as:

$$x_{\text{int},g} = Q_g(x_g) = \text{clamp}\left(\left\lfloor\frac{x_g - z_g}{s_g}\right\rceil, q_{min}, q_{max}\right) \tag{2}$$

where x_g represents the subset of the tensor belonging to group g; s_g and z_g are the scaling factor and zero-point specific to group g.

GSE-INT (Group Shared Exponents Integer) is an advanced quantization technique that further enhances group quantization by sharing the exponent component within each group while quantizing the mantissa under the constraint of the shared exponent [30]. Mathematically, each floating point value *x* is decomposed as:

$$x = (-1)^s \cdot 2^e \cdot m \tag{3}$$

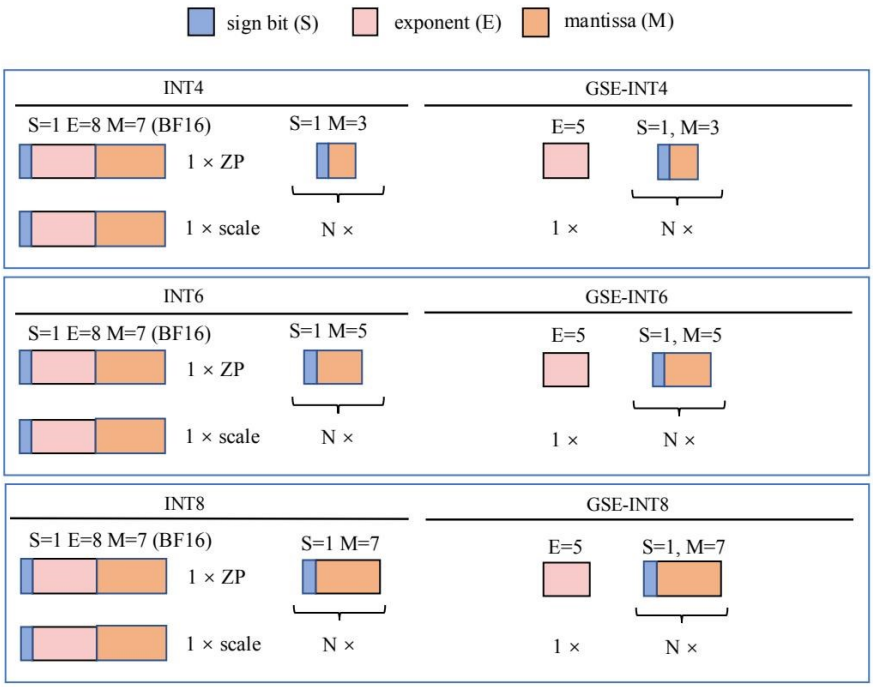

Fig.1. Comparison of quantization formats (INT4, INT6, INT8) with their GSE-INT counterparts (GSE-INT4, GSE-INT6, GSE-INT8)

where s is the sign bit ($s = 0$ for positive values, $s = 1$ for negative values); e is the shared exponent of the group g, typically chosen as the maximum exponent within the group to avoid overflow; m is the mantissa, which is quantized under the constraint of the shared exponent e.

As shown in Figure 1, GSE-INT reduces the memory overhead associated with storing unique exponents for each value, particularly in deep neural networks with large weight tensors.

In GSE-INT, the shared exponent e is determined by the maximum value in the group to ensure that all values within the group can be represented without overflow [5]. The mantissa m is then quantized to a fixed bit-width, allowing for efficient storage and computation. This strategy achieves a balance between memory efficiency and model accuracy by grouping tensor elements with similar magnitudes [2] and sharing their exponents. We use 5 bits for exponent in each group, which is significantly smaller than the classical INT quantization using 16 bits to store scale and zero point.

3.2 GSE in MobileNetV4

In this section, we present the method, GSE-MN4, which involves the integration of Group-Shared Exponents Integer (GSE-INT) quantization into MobileNetV4.
Our approach consists of three steps: (1) layer-wise analysis to determine the quantization sensitivity of each layer, (2) automatic bit-width allocation based on sensitivity thresholds, and (3) group-wise quantization using GSE-INT.

Step 1: Layer Sensitivity Analysis. To effectively apply quantization to MobileNetV4, we first perform a layer-wise sensitivity analysis to identify the sensitivity of each layer to quantization errors. This step is crucial for determining the optimal bit-width allocation for each layer.

Procedure 1. We select a small calibration dataset (e.g., 128 samples) from the training dataset. 2. For each layer l, we quantize its weights and activations to different bit-widths b (e.g. 2-bit, 4-bit, 8-bit) while keeping all other layers at their original precision. 3. Inference using quantized network on the calibration dataset to get $\widehat{O_b^l}$ for each layer l and its bit-width b. 4. We measure the Mean Squared Error (MSE) between the original logits O and the quantized logits $\widehat{O_b^l}$. The sensitivity metric $\widehat{O_b^l}$ for layer l at bit-width b is defined as:

$$S_b^l = MSE\left(O, \widehat{O_b^l}\right) \tag{4}$$

where O represents the original logits, and $\widehat{O_b^l}$ denotes the logits obtained when layer l is quantized to b-bit precision.

In Figure 2, we present the sensitivity plots for MobileNetV4 variants, including MobileNetV4-Conv-Small, MobileNetV4-Conv-Medium, and MobileNetV4-Hybrid-Medium, under INT4, INT6, and INT8 quantization. The plots illustrate the sensitivity metric S_b^l for each layer across different bit-widths, providing a comprehensive analysis of quantization sensitivity.

Step 2: Automatic Bit-Width Allocation. We propose an automatic bit-width allocation strategy based on the sensitivity analysis to assign the optimal bit-width to each layer. This strategy ensures a balance between model compression and accuracy preservation.

Procedure 1. We define a threshold τ that represents the maximum acceptable sensitivity for a layer. 2. For each layer l, we select the minimum bit-width b_l that satisfies $S_b^l < \tau$. 3. This results in a mixed-precision quantization scheme, where sensitive layers are assigned higher bit-widths (e.g., 8-bit) to minimize information loss, while less sensitive layers are assigned lower bit-widths (e.g., 2-bit or 4-bit) to enhance compression efficiency.

The bit-width allocation process is visualized in Figure Y, which demonstrates how the sensitivity metric S_b^l guides the selection of optimal bit-widths for each layer. We identify the minimum bit-width that satisfies the accuracy constraint by comparing S_b^l against the threshold τ.

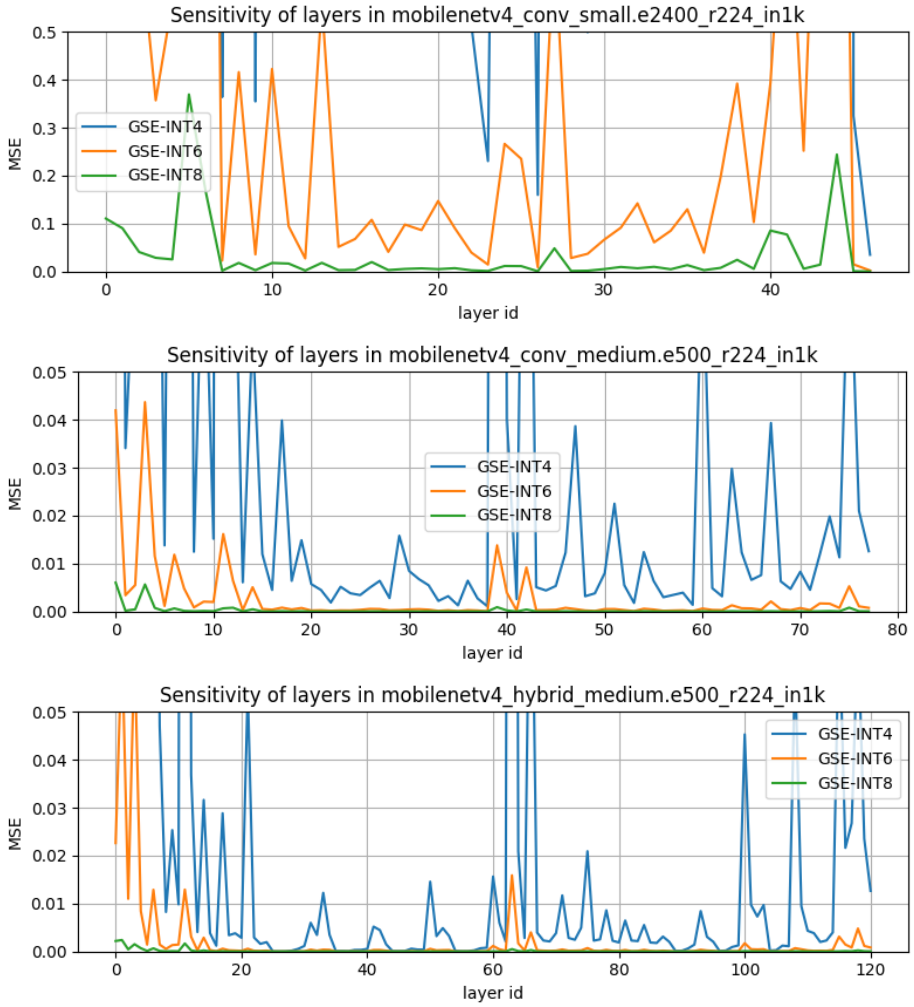

Fig. 2. Layer sensitivity analysis of different MobileNetV4 variants. The group size is 32.

Group-Wise Quantization with GSE-INT. After determining the bit-width b_l for each layer l, we apply group-wise quantization using the Group-Shared Exponents Integer (GSE-INT) method. This approach further optimizes the quantization process by leveraging shared exponents within groups of weights.

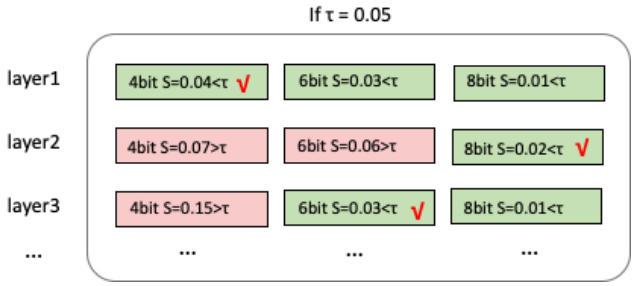

Fig. 3. Bit-width allocation process based on layer sensitivity analysis.

Procedure 1. For each layer l, we partition its weights into groups g based on a predefined strategy (e.g., by output channels or feature map blocks); 2. Within each group g, we represent the weights using the GSE-INT format:

$$x = (-1)^s \cdot 2^e \cdot m \tag{5}$$

where e is the shared exponent in a group, typically derived from the maximum absolute value in the group.

Step 3: The mantissa m is quantized according to the allocated bit-width b:

$$m = \frac{\text{clamp}\left(\left\lfloor \frac{2^{b-1}x}{(-1)^s 2^e} \right\rfloor, 0, 2^{b-1}-1\right)}{2^{b-1}} \tag{6}$$

This ensures that the quantization error is minimized while adhering to the bit-width constraints.

4 Experiments

4.1 Settings

Datasets and Evaluation Metrics. In our experiments, we utilize the ImageNet-1K [3] dataset, which is a widely recognized benchmark for image classification tasks. This dataset comprises 1.28 million training images and 50,000 validation images, spanning 1,000 distinct classes. To evaluate the performance of our proposed GSE-MN4 model, we employ several key metrics, including Top-1 and Top-5 accuracy, which measure the model's ability to correctly identify the most probable class and the top five probable classes, respectively. Additionally, we assess the model's memory footprint, which is crucial for deployment on resource-constrained devices.

Table 1. Quantization results of different methods on MobileNetV4. The group size for GSE-INT is 32, and the group size for INT is 128.

Network	Quantization Method	Bitwidth	Memory (MB)	Top-1 (%)	Top-5 (%)
MN4-Conv-S	BF16	16	7.15	73.74	91.43
	INT8	8	3.69	73.44	91.20
	INT6	6	2.79	51.97	75.94
	GSE-INT8	8	3.64	73.12	91.10
	GSE-INT6	6	2.75	43.09	67.85
	GSE-MIX (τ=0.1)	4/6/8	3.23	73.34	91.17
	GSE-MIX (τ=0.5)	4/6/8	2.73	72.60	90.93
MN4-Conv-M	BF16	16	18.40	79.09	94.77
	INT8	8	9.49	79.09	94.72
	INT6	6	7.19	77.55	93.97
	GSE-INT8	8	9.38	79.02	94.74
	GSE-INT6	6	7.08	77.76	94.07
	GSE-MIX (τ=0.1)	4/6/8	7.20	78.55	94.41
	GSE-MIX (τ=0.5)	4/6/8	6.19	77.55	93.90
MN4-Hybrid-M	BF16	16	20.96	80.43	95.38
	INT8	8	10.81	80.36	95.27
	INT6	6	8.19	79.69	94.92
	GSE-INT8	8	10.69	80.41	95.38
	GSE-INT6	6	8.07	77.74	93.78
	GSE-MIX (τ=0.1)	4/6/8	7.86	79.92	95.14
	GSE-MIX (τ=0.5)	4/6/8	7.03	79.04	94.78

4.2 ImageNet-1K Accuracy Experiments

Baseline Methods. To establish a comprehensive comparison, we evaluate the performance of our proposed GSE-MN4 model against several baseline quantization methods applied to different variants of MobileNetV4 [19]. Specifically, we consider the following networks and quantization methods: (1)MN4-Conv-S: A small convolutional variant of MobileNetV4; (2) MN4-Conv-M: A medium convolutional variant of MobileNetV4; (3) MN4-Hybrid-M: A medium hybrid variant of MobileNetV4, combining convolutional and attention-based layers. We quantize these networks using different quantization methods, including classical group integer quantization to 8 bits and 6 bits (represented as INT8 and INT6, respectively). The GSE-INT8 and GSE-INT6 methods involve quantization with the same bitwidth for all layers. GSE-MIX is the proposed method with different sensitivity thresholds τ.

The results of our experiments are summarized in Table 1, which presents the Top-1 and Top-5 accuracy, as well as the memory footprint for each combination of network and quantization methods.

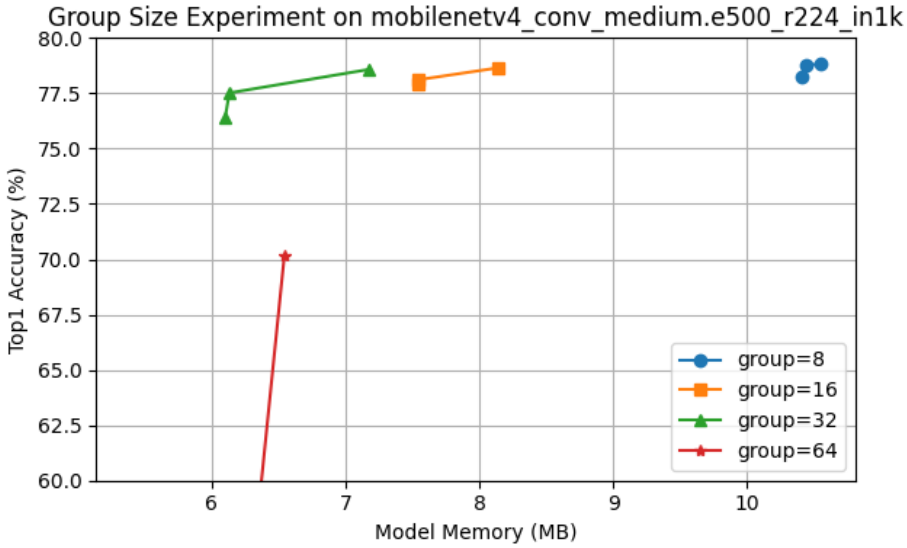

Fig. 4. Influence of Group Size on Model Performance.

Analysis. The proposed GSE-MN4 model, particularly with the GSE-MIX quantization method, demonstrates competitive performance across all network variants. For MN4-Conv-S, GSE-MIX ($\tau = 0.1$) achieves a Top-1 accuracy of 73.34%, with a memory footprint of 3.23 MB. This performance is notably close to the full-precision BF16 model while significantly reducing memory usage.

Similarly, for MN4-Conv-M and MN4-Hybrid-M, the GSE-MIX methods consistently outperform traditional INT8 and INT6 quantization in terms of accuracy, with only a marginal increase in memory footprint. For instance, GSE-MIX ($\tau = 0.01$) on MN4-Hybrid-M achieves a Top-1 accuracy of 78.55%, compared to 77.55% for INT6 quantization. They have nearly the same memory consumption (7.20 MB v.s. 7.19 MB).

These results highlight the efficacy of the Group-Shared Exponents (GSE) approach in balancing accuracy and model efficiency, making it a promising candidate for deployment in mobile and edge devices. The mixed-precision GSE-MIX methods, in particular, offer a superior trade-off between accuracy and compression, demonstrating the importance of adaptive quantization strategies in modern neural network deployment.

4.3 Ablation

Impact of Group Size on Model Performance. To further investigate the impact of group size on the performance of the GSE-MN4 model, we conduct an ablation study focusing on the MobileNetV4-Conv-Medium variant. Specifically, we compare the Top-1 accuracy and memory footprint for different group sizes: 8, 16, 32, and 64. The results are visualized in Figure 4.

As shown in Figure 4, the group size significantly influences both the model's accuracy and memory footprint. The following observations can be made: The ablation study reveals that the choice of group size is crucial in balancing the trade-off between model accuracy and memory efficiency. Smaller group sizes (e.g., 8) tend to prioritize accuracy but demand more memory, while larger group sizes (e.g., 64) considerably reduce memory usage at the expense of accuracy. Group size 32 achieves a Pareto optimal balance between memory usage and top-1 accuracy. It provides a satisfactory compromise, ensuring a reasonable level of accuracy while maintaining a manageable memory footprint. These insights are instrumental in optimizing the GSE-MN4 model for specific deployment scenarios, particularly in resource-constrained environments.

5 Conclusion

In this work, we introduced GSE-MN4: Group-Shared Exponents Integer Quantization for MobileNetV4, a novel quantization framework designed to address the challenges of deploying deep learning models on resource-constrained edge devices. By leveraging the Group-Shared Exponents (GSE) format, our approach significantly reduces the memory footprint while maintaining high accuracy. These results outperform traditional quantization methods, highlighting the potential of GSE-INT for efficient deployment on mobile and edge devices. Our work paves the way for more efficient and accurate deep learning models in resource-constrained environments.

References

1. Choi, J., Wang, Z., Venkataramani, S., Chuang, P.I.J., Srinivasan, V., Gopalakrishnan, K.: Pact: Parameterized clipping activation for quantized neural networks. arXiv preprint arXiv:1805.06085 (2018)
2. Choukroun, Y., et al.: Gradient-based deep neural network quantization. In: CVPR (2022)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
4. Esser, S.K., McKinstry, J.L., Bablani, D., Appuswamy, R., Modha, D.S.: Learned step size quantization (2019). CoRR, abs/1902.08153 4 (2020)
5. Esser, S.K., et al.: Learned step size quantization. In: ICLR (2020)
6. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149 (2015)
7. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1314–1324 (2019)
8. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
9. Jung, S., et al.: Learning to quantize deep networks by optimizing quantization intervals with task loss. In: CVPR (2019)

10. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference. In: ICLR (2018) 12 F. Author et al.
11. Lee, J., et al.: Group-based quantization for efficient neural networks. In: CVPR (2021)
12. Li, Y., et al.: Understanding and overcoming quantization challenges in deep learning. NeurIPS (2020)
13. Lin, J., et al.: Mcunet: Tiny deep learning on iot devices. NeurIPS (2020)
14. Liu, Y., Xu, X., et al.: Edge ai for smart healthcare: Challenges and opportunities. IEEE IoT Journal (2022)
15. Liu, Z., Wang, Y., Han, K., Zhang, W., Ma, S., Gao, W.: Post-training quantization for vision transformer. Advances in Neural Information Processing Systems 34, 28092–28103 (2021)
16. Micikevicius, P., Stosic, D., Burgess, N., Cornea, M., Dubey, P., Grisenthwaite, R., Ha, S., Heinecke, A., Judd, P., Kamalu, J., et al.: Fp8 formats for deep learning. arXiv preprint arXiv:2209.05433 (2022)
17. Nagel, M., Amjad, R.A., Van Baalen, M., Louizos, C., Blankevoort, T.: Up or down? adaptive rounding for post-training quantization. In: International conference on machine learning. pp. 7197–7206. PMLR (2020)
18. Nagel, M., Fournarakis, M., Amjad, R.A., Bondarenko, Y., Van Baalen, M., Blankevoort, T.: A white paper on neural network quantization. arXiv preprint arXiv:2106.08295 (2021)
19. Qin, D., Leichner, C., Delakis, M., Fornoni, M., Luo, S., Yang, F., Wang, W., Banbury, C., Ye, C., Akin, B., et al.: Mobilenetv4: universal models for the mobile ecosystem. In: European Conference on Computer Vision. pp. 78–96. Springer (2024)
20. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
21. Shang, Y., Yuan, Z., Wu, Q., Dong, Z.: Pb-llm: Partially binarized large language models. arXiv preprint arXiv:2310.00034 (2023)
22. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: Vision and challenges. IEEE internet of things journal 3(5), 637–646 (2016)
23. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML (2019)
24. Wang, K., Liu, Z., Lin, Y., Lin, J., Han, S.: HAQ: Hardware-aware automated quantization with mixed precision. arXiv preprint arXiv:1811.08886 (2019), https://arxiv.org/abs/1811.08886
25. Wang, P., Chen, Q., He, X., Cheng, J.: Towards accurate post-training network quantization via bit-split and stitching. In: International Conference on Machine Learning. pp. 9847–9856. PMLR (2020)
26. Xu, M., Yin, W., Cai, D., Yi, R., Xu, D., Wang, Q., Wu, B., Zhao, Y., Yang, C., Wang, S., et al.: A survey of resource-efficient llm and multimodal foundation models. arXiv preprint arXiv:2401.08092 (2024)
27. Yuan, Z., Xue, C., Chen, Y., Wu, Q., Sun, G.: Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In: Computer Vision ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII. pp. 191–207. Springer (2022)
28. Zhang, Y., et al.: Recent advances in neural network quantization. ACM Computing Surveys (2023)
29. Zhao, R., Hu, Y., Dotzel, J., De Sa, C., Zhang, Z.: Improving neural network quantization without retraining using outlier channel splitting. In: International conference on machine

learning. pp. 7543–7552. PMLR (2019) GSE-MN4: Group-Shared Exponents Integer Quantization for MobileNetV4 13

30. Zhou, S., Wang, S., Yuan, Z., Shi, M., Shang, Y., Yang, D.: Gsq-tuning: Groupshared exponents integer in fully quantized training for llms on-device fine-tuning (2025), https://arxiv.org/abs/2502.12913