Article Information

Title	Deadline-Driven CNN Model Based on Residual Quantization
Authors	Ali Haider ALVI, Yoichi TOMIOKA, Yuichi OKUYAMA, and Jungpil SHIN
Citation	International Conference on Electronics, Information, and Communication (ICEIC), Osaka, Japan, 2025, pp. 1-4, doi: 10.1109/ICEIC64972.2025.10879761.
Copyright	
Note	This is the author's accepted version of the paper published in *Proceedings of 2025 International Conference on Electronics, Information, and Communication (ICEIC)*. The final version is available at IEEE Xplore DOI: https://doi.org/10.1109/ICEIC64972.2025.10879761

Deadline-Driven CNN Model Based on Residual Quantization

Ali Haider ALVI, Yoichi TOMIOKA, Yuichi OKUYAMA, and Jungpil SHIN
Graduate School of Computer Science and Engineering
The University of Aizu
Aizuwakamatsu, Japan
{m5281002,ytomioka,okuyama,jpshin}@u-aizu.ac.jp

***Abstract*—In hard real-time systems with strict timing constraints, completing inference within the given deadline is crucial. Model compression methods, such as quantization, have been proposed to reduce inference time. However, these methods do not guarantee that inference will meet the deadline and may degrade accuracy due to aggressive compression. This paper presents a novel deadline-driven model that combines residual quantization with dynamic skipping of residual components to meet hard deadlines while maintaining high accuracy. When tested on the CIFAR-10 dataset using the ResNet-20 architecture, the model achieves a 0% violation rate for timing constraints and delivers accuracy comparable to models that miss deadlines. Compared to non-deadline-driven models, it provides a flexible solution for real-time deployment in hardware-constrained environments, ensuring reliable performance under varying timing demands and resource availability.**

I. Introduction

Deep neural networks (DNNs) have become very popular and have shown impressive results in different areas like computer vision, natural language processing, and speech recognition. They are widely used in real-time applications such as autonomous driving, robotics, and healthcare systems. AI inference performance is constrained in embedded devices due to power consumption and size limitations. Running complex AI models efficiently becomes challenging, affecting both the speed and accuracy of AI-driven tasks on these devices. For example, in a self-driving car, the system must rapidly make accurate decisions based on camera input to avoid obstacles, adapting quickly to varying time constraints due to changing traffic conditions to ensure safety. Considering how deep neural networks in embedded systems are increasingly being used in resource-strapped environments, model compression approaches like model quantization have to be employed. Among other reasons, this helps to save memory space and energy without compromising the accuracy of the network.

Quantizing the parameters and activations of a neural network to lower bit-widths allows the arithmetic units of a neural network accelerator to be made smaller, enabling higher parallelism within the same area. As a result, low-bit quantization is effective in reducing inference time. Additionally, some Central Processing Units (CPUs) and Graphics Processing Units (GPUs) can achieve higher computational performance in low-bit arithmetic by subdividing a high-bit arithmetic unit into multiple low-bit units. For instance, the NVIDIA T4 offers a peak performance that is four times greater for 4-bit integer arithmetic (INT4) compared to 16-bit floating-point arithmetic (FP16). Because some GPUs support INT4 operations, 4-bit quantization is a good option for achieving faster speeds at the cost of a slight reduction in accuracy.

As we discussed in the next section, many studies [1]–[3], [6]–[10], [12], [13] have presented quantization-based acceleration. In the existing quantization approach, edge AI models are typically compressed offline to meet the timing constraints of edge devices. Even if a model's inference time exceeds the worst-case timing constraint, it may still meet the required timing in most cases. However, there remains a slight risk of timing violations. To guarantee correct operation under worst-case conditions, models need to be compressed through pruning and/or quantization to ensure they meet the worst-case timing constraints. This, however, may lead to a degradation in recognition accuracy and reduce the system's security, even though, in many cases, a longer inference time would be acceptable. Our goal is to achieve higher average accuracy in hard real-time systems by utilizing the additional available time beyond the worst-case scenario while still ensuring strict deadlines are met.

In this paper, we take a step beyond most modeling approaches, called deadline-driven models, which involve skipping multiple layers within a residual quantized model. In contrast to existing quantized models, our method introduces a dynamic and adaptive mechanism that significantly improves system efficiency by skipping computations with low contributions during inference to meet a given deadline. This adaptability is particularly valuable in resource-constrained environments, such as hard real-time embedded systems, where system requirements can fluctuate in response to changing conditions. Furthermore, our method does not require model retraining, ensuring scalability across diverse architectures. In our experiments, we demonstrate that 4-bit residual quantization-based deadline-driven models achieve comparable accuracy to their floating-point counterparts while ensuring that given deadlines are met.

II. Literature Review

Low-bit quantization, especially 4-bit, is crucial for optimizing deep neural networks in resource-constrained environments like edge devices, providing good speed, power

efficiency, and accuracy. It has shown promising results in image recognition, where it speeds up processing and reduces hardware requirements with minimal accuracy loss. For instance, Trusov et al. achieved 48% acceleration in 4-bit CNNs on ARM-based mobile devices [7], while Xu et al. developed STQN to minimize accuracy loss in sub-4-bit models [10]. DyBit further improved accuracy by 2% and achieved 8.1x acceleration by adjusting floating-point bit widths [13]. Similarly, Xia et al. introduced UPoT Quantization to enhance CNN accuracy by up to 6% while maintaining energy efficiency [9], and Shomron et al. focused on sparsity-aware quantization to boost inference speed [6]. Xu et al. also presented KMDFQ for privacy-sensitive tasks, with only a 1.2% precision loss [10].

Research into Large Language Models (LLMs) also benefits from 4-bit quantization. Dettmers and Zettlemoyer achieved optimal accuracy with 4-bit precision blocks [3], and Ashkboos et al.'s QUIK scheme increased throughput 3.4x for models like LLaMA and Falcon while maintaining high accuracy [1]. SpQR further optimized LLMs with less than 1% performance degradation, making them viable for consumer hardware [2]. Wu et al. demonstrated 4-bit quantization in transformers, though challenges remain with real-time systems [8], and Zhang et al. developed a hardware-friendly method for BERT models [12].

While combining CPUs, GPUs, and specialized circuits enhances low-memory inference, these methods depend on offline compression and cannot guarantee deadlines in real-time multi-task systems. To address this, the paper proposes a deadline-driven model that maximizes inference accuracy within strict time constraints, ensuring timely task completion.

III. Methodology of our Proposed Model

In our proposed method, we employ residual quantization on both the weights and activations. In contrast to prior work, this uses the model inspired by Yvinec et al. [11], which systematically reduces the precision of weights and activations from floating point to integer format by quantizing remaining residual errors at each step.

A. Weight Quantization Process

In the iterative scheme of weight quantization, the relative decrease of precision and residual error is maintained. The process is as follows:

1) **Initial Weight Residual:** The decomposition is straightforward as we involve the follow up of the original weights W so we set the residual as $\Delta W_0 = W$.
2) **Scaling Factor:** In the iteration k, we perform this step by computing the scaling factor α_k, this is done to ensure that the weights are reduced to lower precision weights.
$$\alpha_k = \frac{2^{n-1} - 1}{\max(|\Delta W_{k-1}|)}$$
where n is the number of bits (e.g., $n = 4$).
3) **Quantization:** The residual weight ΔW_{k-1} is quantized using the scaling factor α_k:
$$W_k = \text{round}(\alpha_k \cdot \Delta W_{k-1})$$
4) **Residual Update:** After quantization, the residual error is updated iteratively by reducing the error using the following equation:
$$\Delta W_k = \Delta W_{k-1} - \frac{W_k}{\alpha_k}$$
In every subsequent step, this equation helps to curb the residuals carried from the previous quantization steps. And the processes are carried on until the residual error ΔW_k is less than a chosen tolerance or the specified number of iterations, trying to minimize the error that is induced by quantization.

After all iterations, the final quantized weights are the sum of the quantized components over all iterations:
$$W_{\text{final}} \approx \sum_{k=0}^{K} \frac{W_k}{\alpha_k}$$
Here, K is the number of residual quantized parameters. The method for residual quantization for weights is the same residual quantization discussed by Yvinec et al. [11].

B. Activation Quantization Process

As not in the work in [11], we apply residual quantization to activations in addition to weights. The process includes:

1) **Initial Activation Residual:** We start with the original activations X, and initialize the residual $\Delta X_0 = X$.
2) **Scaling Factor:** At each iteration j, calculate the scaling factor β_j:
$$\beta_j = \frac{2^n - 1}{\max(\Delta X_{j-1}) - \min(\Delta X_{j-1})}$$
3) **Offset γ_k:** The offset is calculated to adjust the activation range:
$$\gamma_j = \min(\Delta X_{j-1})$$
4) **Quantization:** The activation residual ΔX_{j-1} is quantized:
$$X_k = \text{round}(\beta_j \cdot (\Delta X_{j-1} - \gamma_j))$$
Note that broadcasting rules are assumed for operations with tensor data ΔX_{j-1} and scalar γ_j.
5) **Residual Update:** As with weights, we iteratively reduce the residual error for activations with the following update equation:
$$\Delta X_j = \Delta X_{j-1} - \frac{X_j}{\beta_j} - \gamma_j$$

The final quantized activations are computed after all iterations:
$$X_{\text{final}} \approx \sum_{j=0}^{J} \left(\frac{X_j}{\beta_j} + \gamma_j \right) \tag{1}$$
Here, J is the number of residual quantized activations.

C. Residual Quantized Dot Product

Both Matrix multiplication and convolution consist of the dot product. The dot product of W and X is approximately represented by the weighted summation of dot products W_k and X_j for $(k, j) \in \{1, 2, \ldots, K\} \times \{1, 2, \ldots, J\}$.

$$W \cdot X \approx \sum_{k=0}^{K} \sum_{j=0}^{J} \frac{W_k \cdot X_j}{\alpha_k \cdot \beta_j} + \Gamma \tag{2}$$

Here, Γ is a constant calculated by w_k and γ_j.

D. Sensitivity Analysis and Skipping Order

After applying residual quantization, each convolutional layer $Conv(W, X)$ is transformed into the weighted summation of $K \times J$ residual quantized components $Conv(W_k, X_j)$. The basic idea of realizing a deadline-driven model is skipping the low-impact components according to a given deadline. To understand the impact of individual components, we performed a sensitivity analysis:

1) **Initial Accuracy:** First, we measured the model's original accuracy with no residual components skipped.
2) **Skip Residuals:** For each residual component $Conv(W_k, X_j)$, we skipped it and re-evaluated the accuracy.
3) **Calculate Sensitivity:** The sensitivity of each residual quantized component is defined as the accuracy reduction from the original accuracy when it is skipped.
4) **Skipping order:** The deadline-driven model completes inference within the given deadline by skipping the minimum number of residual quantized components with the lowest sensitivity.

Note that we must not skip the most sensitive residual quantized component in each layer not to break the path from the primary input to the primary output.

IV. Experimental Results

A. Model and Dataset

In order to verify the possibility of improving accuracy and safety using a deadline-driven model based on residual quantization, this paper evaluated using ResNet-20 [4] model. The ResNet-20 is converted to a deadline-driven model based on residual quantization. For evaluation, we extend the CIFAR-10 [5] dataset by assigning timing constraints to each image with two scenarios. The inference time is strongly related to the computational cost, that is, the number of operations. Therefore, we assign computational cost constraint to each image. In the first (second) scenario, the computational cost constraint ranges from 61 MOPs to 170 (80) MOPs.

B. Results of Residual quantization

We then utilize residual quantization with respect to various bit-width combinations of weights (W) and activations (A) as 8W-8A, 6W-6A, 4W-4A, and 2W-2A.

Our floating-point ResNet-20 achieved an accuracy of 0.9176. As shown in Table I, residual quantized models achieved comparable accuracy with the floating-point model

TABLE I
PERFORMANCE OF RESIDUAL QUANTIZED RESNET-20 ACROSS DIFFERENT BIT-WIDTH CONFIGURATIONS. THE CORRESPONDING FLOATING-POINT MODEL ACHIEVES AN ACCURACY OF 0.9176.

Bits	**8W, 8A**	**6W, 6A**	**4W, 4A**	**2W, 2A**
Iterations (K,J)	(2 , 2)	(2 , 2)	(2 , 2)	(10 , 4)
Accuracy	0.9176	0.9176	0.9174	0.9175

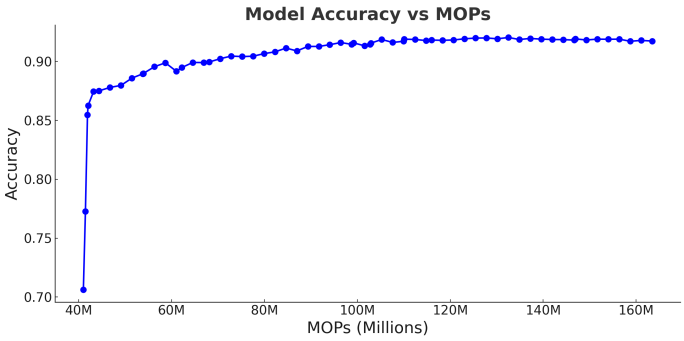

Fig. 1. Trade-off curve illustrating the relationship between MOPs and model accuracy.

across all quantization configurations The 2W–2A configuration requires more iterations (10 for weights, 4 for activations) than higher bit-widths, highlighting a trade-off between computational effort and precision. On the other hand, The 4W-4A requires only four residual quantized components. Although it requires four times more operations than FP16 model, some GPU, such as NVIDIA T4 supporting INT4 operations, can achieve the four times peak performance for INT4 operations than FP16 operations. Therefore, it is expected that residual quantized model with 4W-4A is as fast as FP16 model.

C. Trade-off Curve: Model Constraints and Performance

The trade-off curve illustrates the relationship between the computational cost (in terms of Millions of Operations, or MOPs) and the model's accuracy for residual quantized ResNet-20 with 4W-4A when we skip the residual quantized components according to the skipping order described in Section III-D. As shown in the graph, the model's accuracy improves as the number of operations (the number of residual quantized components) increases. Our idea is to allow an accuracy penalty due to computational cost reduction only for images to which tight computational cost constraints are assigned.

D. Violation Rate Calculation

If the number of operations required for a single inference is greater than the constraint assigned to an image, it is regarded as a timing violation. The violation rate is a metric that measures how often a model violates the computational cost constraints of images. The proposed model never violates the deadline if any computational cost constraint is greater than or equal to the minimum number of operations required for a single inference of the deadline-driven model.

TABLE II
PERFORMANCE COMPARISON WITH COMPUTATIONAL COST CONSTRAINTS OF 61–80 MOPS.

Model	#operations	Accuracy	Violation Rate
Proposed	61–170 MOPs	91.32%	0.00%
Conventional (Fast)	68 MOPs	89.96%	5.91%
Conventional (Slow)	163 MOPs	91.74%	93.67%

TABLE III
PERFORMANCE COMPARISON WITH COMPUTATIONAL COST CONSTRAINTS OF 61–80 MOPS.

Model	#operations	Accuracy	ViolationRate
Deadline Driven	61–170 MOPs	90.08%	0.00%
Conventional (Fast)	68 MOPs	89.96%	37.47%
Conventional (Slow)	163 MOPs	91.74%	100.00%

E. Comparison with conventional models

In terms of the accuracy and violation rate, we compared the deadline-driven ResNet-20 with 4W-4A residual quantization with fast and slow non-deadline-driven models whose computational costs are 68 MOPs and 163 MOPs, respectively. They correspond to two points in Fig. 1. We summarize the results for the first dataset with computational cost constraints of 61 MOPs to 170 MOPs in Table II.

Although the Slow model achieved 91.74% accuracy, the violation rate was 93.67% violation rate. Although the Fast model reduced the violation rate to 5.91%, the accuracy was reduced to 89.96%. In contrast, the proposed Deadline-Driven model dynamically adjusts computational cost, ensuring 0% violations while maintaining 91.32% accuracy.

We summarized the results for the second dataset with computational costs of 61-80 MOPs in Table III. This dataset imposes more stringent limits, highlighting each model's ability to adjust to varying computational requirements.

Even a fast model exceeds the constraint for 37.47% of the images, though it maintains an accuracy of 89.96%. On the other hand, the deadline-driven model achieves a 0% violation rate by adjusting the computational cost dynamically. The accuracy is also comparable with that of the fast model.

Across both datasets, our proposed Deadline-Driven model outperforms the conventional models in terms of violation rate, consistently achieving a 0% violation rate. While the Slow model achieves slightly higher accuracy, its lack of flexibility results in frequent deadline violations, making it unsuitable for real-time applications. The Fast model has a lower violation rate than the Slow model, but it still struggles with meeting constraints for many test images. In contrast, the Deadline-Driven model strikes a balance between accuracy and computational efficiency by dynamically adjusting computational cost to meet the constraints for each test image, making it the most versatile and effective solution for hard real-time scenarios.

V. CONCLUSION

In this paper, we presented a novel deadline-driven model with dynamic residual component skipping, tailored for resource-constrained environments. With an extended CIFAR-10 dataset and ResNet-20 model, we show that our proposed model has a 0% violation rate and is competitive with respect to accuracy, outperforming conventional static models. The model dynamically adjusts computational complexity in real time while this achieves an effective balance between accuracy and efficiency, it makes the model well-suited for hard-real time embedded systems. In future work, this approach would be extended to more complex models and various datasets, and it would be deployed in real-world scenarios, including autonomous systems and IoT devices.

ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Number JP23H03477.

REFERENCES

[1] S. Ashkboos, I. Markov, E. Frantar, T. Zhong, X. Wang, J. Ren, T. Hoefler, and D. Alistarh. Towards end-to-end 4-bit inference on generative large language models. *arXiv preprint arXiv:2310.09259*, 2023.

[2] Tim Dettmers, Roman Svirschevski, Vahram Egiazarian, Dmitry Kuznedelev, Elias Frantar, Sina Ashkboos, Alexey Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*, 2023.

[3] Tim Dettmers and Luke Zettlemoyer. The case for 4-bit precision: k-bit inference scaling laws. *arXiv preprint arXiv:2212.09720*, 2022.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[5] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[6] Gil Shomron, Fadi Gabbay, Samer Kurzum, and Uri Weiser. Post-training sparsity-aware quantization. *arXiv preprint arXiv:2105.11010*, 2021.

[7] Alexander Trusov, Ekaterina Limonova, Dmitry Slugin, Denis Nikolaev, and Victor Arlazarov. Fast implementation of 4-bit convolutional neural networks for mobile devices. *arXiv preprint arXiv:2009.06488*, 2021.

[8] Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. Understanding int4 quantization for language models: Latency speedup, composability, and failure cases. *arXiv preprint arXiv:2301.12017*, 2023.

[9] Tong Xia, Bo Zhao, Jun Ma, Guozhen Fu, Wenye Zhao, Nan Zheng, and Pengcheng Ren. An energy-and-area-efficient cnn accelerator for universal powers-of-two quantization. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 70:1242–1255, 2023.

[10] Wenqing Xu, Feng Li, Yanfei Jiang, Andrew Yong, Xiangjian He, Peipei Wang, and Jian Cheng. Improving extreme low-bit quantization with soft threshold. *IEEE Transactions on Circuits and Systems for Video Technology*, 33:1549–1563, 2023.

[11] Edouard Yvinec, Arnaud Dapgony, Matthieu Cord, and Kevin Bailly. Rex: Data-free residual quantization error expansion, 2023.

[12] J. Zhang, Y. Zhang, H. Dong, and W. Zhang. Clipping and piece-wise quantization for 4-bit bert inference. *arXiv preprint arXiv:2206.05762*, 2022.

[13] Jiaxin Zhou, Jiacheng Wu, Yuan Gao, Yuchen Ding, Chunhua Tao, Baoyuan Li, Fangzhou Tu, Kaisheng Cheng, and Ngai Wong. Dybit: Dynamic bit-precision numbers for efficient quantized neural network inference. *arXiv preprint arXiv:2302.12510*, 2023.