Layer-wise Quantization in Green-Aware AI*

Farwa Ikram[1,*,†], Dipanwita Thakur[1,†], Antonella Guzzo[1] and Giancarlo Fortino[1]

[1]The University of Calabria, Arcavacata di Rende, Italy

Abstract

Deep neural networks (DNN) have become integral to many artificial intelligence (AI) applications due to their superior performance across a wide range of tasks. However, their deployment in edge devices is limited by high computational and energy demands, which also increase carbon emissions. Quantization, particularly layer-wise quantization, has emerged as a promising technique to mitigate these limitations by reducing numerical precision and computational overhead. In this paper, we present a comprehensive study on the effects of some layer-wise quantization strategies on the energy efficiency and the environmental impact of DNNs, with a specific focus on green-aware AI. We examine uniform, ascending, and descending quantization schemes to understand how per-layer bit-width assignments can be optimized for minimal energy usage while preserving model accuracy. Our analysis includes a quantitative assessment of energy consumption and associated CO_2 emissions. Experiments conducted on standard benchmarks, ResNet18 trained on CIFAR-10 and CIFAR-100, demonstrate that our method achieves up to a 45% reduction in energy consumption and memory usage with competitive degradation in accuracy, offering a practical solution for sustainable AI deployment.

Keywords

Green AI, Layer-wise quantization, Energy Efficiency, Carbon Emission

1. Introduction

Deep neural networks (DNNs) have revolutionized the field of machine learning, enabling breakthroughs in computer vision [1], natural language processing [2], and autonomous systems [3]. Despite their success, the computational intensity and memory footprint of DNNs have raised significant concerns, especially for deployment on resource-constrained edge devices such as mobile phones, embedded systems, and IoT devices [4, 5]. Moreover, the environmental cost of training and deploying large-scale DNNS, including energy consumption and CO_2 emissions has drawn increasing scrutiny for both academia and industry due to sustainable development goals (SDGs) defined by the United Nations [6, 7].

Quantization is a well established technique for reducing the complexity of DNNs by lowering the precision of weights and activations [8, 2]. While traditional quantization approaches typically apply a uniform bit-width across all layers [9], recent research suggests that a more nuanced, layer-wise strategy may better balance accuracy and efficiency. This paper explores the design and impact of some layer-wise quantization schemes, with a particular emphasis on green-aware computing, i.e., reducing the environmental impact of AI systems without sacrificing performance.

We investigate three layer-wise quantization strategies: uniform quantization, ascending quantization (lower to higher bit-width from shallow to deeper layers), and descending quantization (higher to lower bit-width). Through empirical analysis and benchmarking, we evaluate their effects on model performance, memory usage, energy consumption, and CO_2 emissions. Our work provides new insights into how intelligent bit-width allocation can yield more sustainable AI solutions.

2nd Workshop on Green-Aware Artificial Intelligence, 28th European Conference on Artificial Intelligence (ECAI 2025), October 25–30, 2025, Bologna, Italy

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

*Corresponding author.

†These authors contributed equally.

farwa.ikram@dimes.unical.it (F. Ikram); dipanwita.thakur@unical.it (D. Thakur); antonella.guzzo@unical.it (A. Guzzo); giancarlo.fortino@unical.it (G. Fortino)

1.1. Motivation

The motivation of this work stems from the growing need to reconcile the performance of deep learning models with environmental and deployment constraints. Many real-world applications require the deployment of DNNs on edge devices, which are limited in terms of computational power, memory, and battery life. Reducing the energy and memory demands of models is essential for feasible and responsive edge deployment. Training and deploying DNNs have a non-trivial environmental footprint. Studies show that model inference at scale contributes significantly to carbon emissions. Developing quantization strategies that minimize energy use can directly contribute to reducing this impact. While uniform quantization offers simplicity, it may not be optimal. Different layers in a network have varying sensitivity to quantization, suggesting that intelligent, layer-specific strategies could yield better trade-offs between efficiency and accuracy [9]. Previous work has not sufficiently explored the impact of different layer-wise quantization strategies on green metrics such as energy consumption and CO_2 emissions. This study fills that gap by providing a systematic comparison and practical insights.

1.2. Contribution

This paper makes the following key contributions:

- We systematically evaluate three distinct layer-wise quantization approaches, such as uniform, ascending, and descending, analyzing their trade-offs in terms of energy efficiency, memory usage, and classification accuracy. We also analyze the energy and environmental impact of layer-wise quantization strategies in DNNs, focusing on reducing CO_2 emissions while maintaining model performance.
- We present a quantitative assessment of energy savings and corresponding reductions in CO_2 emissions resulting from each quantization strategy, enabling a clearer understanding of their environmental benefits.
- Extensive experiments on two different flavors of ResNet18 using CIFAR-10 and CIFAR-100 demonstrate that our proposed strategies can reduce energy consumption and memory footprint by up to 45%, with competitive degradation in accuracy.

2. Related Work

To achieve the objectives of Green-Aware AI, extensive work has been conducted on layer-wise quantization, which can lead to reduced communication and computing overhead. This method allows for control precision by layer, which allows for fine-tuning trade-offs between memory access, size, and inference costs, resulting in a significant reduction of the environmental impact of deep learning models with little or no decline in performance. In [21], the authors proposed a layer-wise quantization method that assigns different bit-widths to layers based on their importance, using two scoring metrics, one measuring changes in input-output representations and another based on weight distribution. Applied with techniques like GPTQ and Quanto, this approach preserved over 90% accuracy even when compressing models like LLama-13B to an average of 3.25 bits, outperforming uniform quantization and moderate pruning in both classification and reasoning tasks. In [17], a lightweight multilayer perceptron (MLP) model was introduced for speech emotion recognition (SER) with a layer-wise adaptive quantization (LAQ) scheme to compress the size of the model while maintaining accuracy. Their approach applies different bit-widths to layers with scores based on parameter proportion, entropy, and weight variance. In another work [13], Gluska and Grobman proposed a method to identify which layers of a neural network contribute the most to quantization errors by quantizing one layer at a time. Their analysis showed that performance loss is often caused by a small number of layers. Using this insight, they applied targeted fixes such as clipping only the most problematic layer, which improved the accuracy of ResNet26 from 74.28% to 75.91%, outperforming global quantization techniques.

Reference	Model	Method	Bit Allocation Strategy	layer-wise Quantization Basis	Results
[10]	VGG-16, ResNet18	RL-based Mixed-Precision	Descending	Parameter count per layer	11× compression, 93.5% accuracy vs 93.6%
[11]	LLama2	Quantization Error Propagation (QEP)	Mixed	Error sensitivity across layers	Accuracy improved from 67.9% to 70.1% with 2-bit GPTQ
[12]	LLama-13B	Layer-wise Quantization via Importance Scores	Descending	Layer importance via input-output changes and weight distribution	Over 90% accuracy at 3.25 avg bits
[13]	ResNeXt26	Layer-wise Error Analysis	Fixed	Empirical error sensitivity (quantize one layer at a time)	Accuracy improved from 74.28% to 75.91%
[14]	ImageNet	LTNN (Tensorized Decomposition)	Mixed precision per layer	Tensor decomposition structure	Up to 64A compression, 13.95% top-5 error at 35.84A
[15]	WGANs	Adaptive Layer-wise Quantization	Adaptive	Gradient noise model and convergence behavior	Up to 47% faster training as compared to Q-GenX
[16]	VGG19, ResNet18/34	Hybrid Quantization and Pruning	Descending	Statistical metrics (entropy, variance, sparsity)	91% accuracy, avg. 1.08 bit-width, up to 81% sparsity
[17]	TESS, EMODB, SAVEE	Layer-wise Adaptive Quantization for SER	Descending	Parameter proportion, entropy, weight variance	Up to 99.29% accuracy
[18]	LLaMA models	SensiBoost, KurtBoost	Descending	Activation sensitivity and kurtosis	Up to 9% lower perplexity with only 2% more memory
[19]	ResNet18	Edge-MPQ with RISC-V	Mixed precision	Hardware constraints and training sensitivity	47.67A speedup, 6.7% higher PTQ accuracy
[20]	ImageNet (AlexNet)	Entropy-based layer-wise Quantization	Descending	Weight and activation entropy	90.28% accuracy, 45.64% top-1 (ImageNet)

Table 1: Performance analysis of existing layer-wise quantization methods.

Similarly, in [18], a layer-sensitive quantization strategy for large language models that selectively assigns higher precision to layers identified as more sensitive to quantization errors. They developed SensiBoost and KurtBoost to guide memory allocation at the layer level. These methods improved quantization accuracy while staying within a modest memory budget. For instance, SensiBoost reduced perplexity by up to 9% on LLaMA models with only a 2% increase in memory. In [19], a novel hardware, layer-wise mixed-precision quantization (MPQ) framework tailored for edge computing, combining both efficient inference hardware and novel quantization search algorithms. Their design integrates versatile inference units supporting INT2 to INT16 directly into a RISC-V processor pipeline, achieving a speed up of up to 47.67× and energy efficiency of 20.51 TOPS/W over the baseline RV64IMA core. The authors in [20] presented an adaptive layer-wise quantization method that assigns bit-widths to each layer based on its importance, measured using the entropy of weights and activations. To stabilize activation distribution, they applied L2 regularization during training, enabling more effective compression. Their approach outperformed fixed-bit quantization methods in both accuracy and compression efficiency. On CIFAR-10, they achieved 90.28% accuracy with 27.5× compression for weights, and 86.14% accuracy with joint weight-activation quantization. In [10], the authors proposed a reinforcement learning-based

framework to perform mixed-precision quantization across deep neural network layers. Instead of applying a fixed bit-width uniformly across all layers, their method leverages the parameter count of each layer to guide the bit allocation process. By using Deep Q-Learning, the system learns to assign different bit-widths to each layer in a way that balances accuracy with model size reduction. For example, VGG-16 on CIFAR-10 was compressed from 59.91MB to 5.57MB with only a 0. 1% drop in precision (93. 6% to 93. 5%). A novel approach called quantized optimistic dual averaging (QODA) was proposed in [15] for distributed variational inequality problems. Their approach provides theoretical guarantees on both quantization error and communication cost, and generalizes previous global quantization results. When applied in training WGAN on CIFAR-10 and CIFAR-100 among 4 GPUs, their method not only provided better accuracy than Q-GenX baseline but also accelerated the training process by up to 47%, which shows the theoretical and empirical advantages in distributed optimization. A hybrid approach of quantization and pruning via adaptation based upon the statistical significance of the neural network layer was proposed in [16]. The method adaptively optimizes bit width and pruning thresholds on a per-layer basis using measures such as entropy, variance, and sparsity to minimize model size without sacrificing accuracy. For VGG19, ResNet18, and ResNet34 on CIFAR-10, the method maintains accuracy over 91%, and averages bit-width as small as 1.08, and can induce parameter sparsity up to 81%. In [14], a compression technique was developed known as LTNN, which reshapes neural network weights into high-dimensional tensors and applies low-rank tensor train decomposition to employ in a layer-wise implementation for training to save model size and, at the same time, preserve accuracy. Additionally, they suggested quantizing tensor cores for more compression. On MNIST and CIFAR-10 benchmarks, LTNN can outperform state-of-the-art methods with compression and 21.57 x 2.2 loss compression. On ImageNet, they reported 8.8× compression with 13.15% top-5 error, which increased to 13.95% error at 35.84× compression with quantization. We summarize the proposed layer-wise quantization methods in Table 4.

3. Methodology

Quantization plays a crucial role in the efficient deployment of machine learning (ML) models and has consequently become a prominent focus in current research. One widely recognized technique for model compression that effectively reduces computational complexity and storage requirements is model quantization [22]. This approach involves transforming high-precision floating-point numbers, such as 32-bit representations, into lower-precision integers, including formats like 8-bit or 4-bit, and even smaller sizes. While quantization provides significant benefits in terms of computational efficiency and reduced storage demand, it also presents challenges related to precision loss [23]. The sensitivity of various layers within a model to quantization can differ significantly, and conventional fixed-precision quantization methods may adversely affect the performance of critical layers, thereby impacting the model's overall inference capabilities. To mitigate these issues, researchers have been developing mixed-precision quantization techniques. These methods strategically allocate different bit lengths to individual layers, striking a balance between resource utilization and model performance, ultimately enhancing the effectiveness of the quantization process. We analyze the impact of fixed bit layer-wise quantization, as well as the different levels of precision to individual layers of a neural network, allowing for a more fine-grained trade-off between model accuracy and computational efficiency. The graphical representation of explained methodology is shown in Figure 1 .

3.1. Layer-wise Quantization

Layer-wise quantization is a model compression approach that quantizes different layers of a neural network with different numerical precisions. Rather than utilizing an equal bit-width policy across all layers, our solution stands for selective quantization, in which different layers are quantized in a fine-grained manner that corresponds to the importance of the layer's contribution to the overall model's performance. Less important or more robust layers can usually be quantized to a smaller number of bits (2-bit or 3-bit), while sensitive layers are kept at higher precision to avoid loss in accuracy. This technique

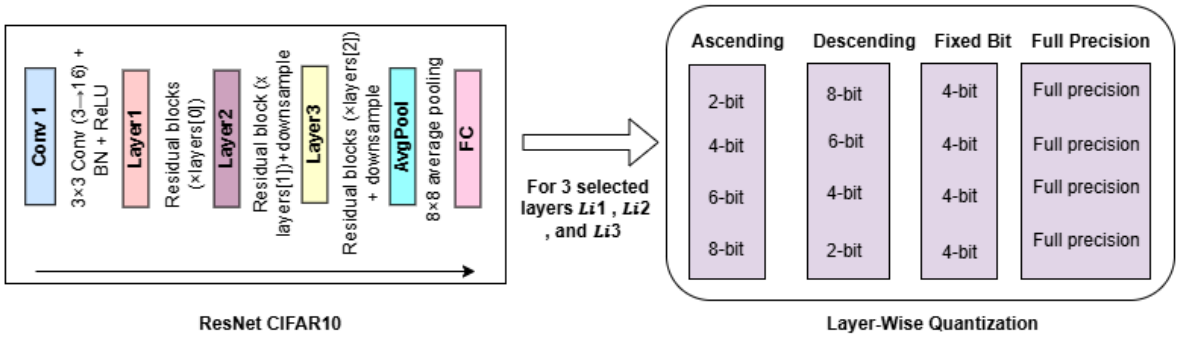


Figure 1: Block Diagram of Methodology

Algorithm 1 layer-wise Quantization Strategy

Step 1: Initialization

Given a neural network N with m layers: $L_1, L_2, \ldots, L_m$

Identify some target layers to quantize: $L_{i_1}, L_{i_2}, L_{i_3}, \ldots, L_{i_n}$

Choose quantization mode: *ascending*, *descending*, or *fixed*

Define bit-width set $B = \{b_1, b_2, b_3, \ldots, b_m\}$ such that $b_1 \leq b_2 \leq b_3 \leq \ldots \leq b_m$

Step 2: Assign Bit-widths

if mode == *ascending* **then**

    Assign bit-widths $\{b_1, b_2, b_3, \ldots, b_n\}$ to layers $\{L_{i_1}, L_{i_2}, L_{i_3}, \ldots, L_{i_n}\}$ in order

else if mode == *descending* **then**

    Assign bit-widths $\{b_n, b_{n-1}, b_{n-2}, \ldots, b_1\}$ to layers $\{L_{i_1}, L_{i_2}, L_{i_3}, \ldots, L_{i_n}\}$ in order

else if mode == *fixed* **then**

    Assign equal bit-width to all targeted layers $\{L_{i_1}, L_{i_2}, L_{i_3}, \ldots, L_{i_n}\}$

end if

Step 3: Quantization

for each layer L in $\{L_{i_1}, L_{i_2}, L_{i_3}, \ldots, L_{i_n}\}$ **do**

    Quantize the weights of L using the assigned bit-width

    Quantize the input and output activations of L using the same bit-width

end for

Step 4: Output

Return the quantized network $\hat{N}$

optimizes the trade-off of computational cost and the model quality by focusing its quantization on the most beneficial places. For example, in convolutional networks, the first or middle layers can tolerate a higher level of aggressive quantization as they're naturally redundant in the extraction of the features, while the later layers are often placed near the decision boundary, which requires higher precision for classification accuracy. The quantization of each layer is conducted layer-wise. A fixed-bit quantization is that all of the selected layers use the same bit-width. Existing adaptive quantization techniques employ ascending-trend, where the quantization level rises as training progresses and descending-trend quantization, where the quantization level decreases as training progresses, in addition to fixed-bit quantization. The algorithm 1 shows the steps of the layer-wise quantization method in which a neural network N consisting of m layers, arranged in topological order as $L_1, L_2, \ldots, L_m$. Each layer typically performs a standard operation such as convolution, fully connected computation, or pooling. In this method, three specific layers, denoted as L_{i_1}, L_{i_2}, and L_{i_3}, are selected for quantization, while the remaining layers are retained in full precision. Lines 5-9 explore three configurations:(1) *ascending quantization*, where layers are quantized using increasing precision levels (2) *descending quantization*,

where decreasing precision is applied across layers, and (3) *fixed-bit quantization*, in which all selected layers are quantized using the same bit-width. Lines 13-14 show application of activation quantization to its weights and its input/output for each quantized layer. Algorithm 1 continues to run until either it converges or satisfies a predefined condition, such as achieving 95% accuracy.

Stage	Operation	Output Shape (CIFAR input: 32×32)
Conv1	3×3 Conv (3→16) + BN + ReLU	[16×32×32]
Layer1	Residual blocks (×layers[0])	[16×32×32]
Layer2	Residual blocks (×layers[1]) + downsample	[32×16×16]
Layer3	Residual blocks (×layers[2]) + downsample	[64×8×8]
AvgPool	8×8 average pooling	[64×1×1]
FC	Fully connected → `num_classes`	[10]

Table 2: Layer-wise architecture of ResNet-CIFAR10 with layer-wise quantization

3.2. Customized ResNet18 for CIFAR-10

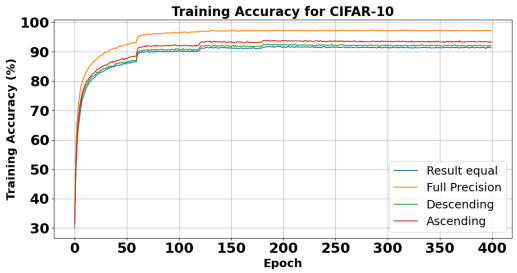
ResNet18 [24] is one of the popular DNN-based model for image classification. The architecture of ResNet18 used for image classification is represented in Table 2, known as ResNet-CIFAR. The ResNet-CIFAR architecture begins with an initial convolutional layer (Conv1) that applies a 33 convolution to the input RGB image, increasing the channel depth from 3 to 16. This is followed by batch normalization and a ReLU activation function, preserving the spatial resolution of 3232. The first major stage, Layer1, consists of multiple residual blocks (as specified by layers[0]) and operates at a constant resolution of 3232, maintaining 16 output channels. Layer2 introduces spatial downsampling via a stride of 2 in the first residual block, reducing the feature map size to 16×16 and increasing the number of channels to 32; it contains layers[1] residual blocks. Similarly, Layer3 performs another downsampling step, reducing the resolution to 88 while increasing the channel count to 64, with layers[2] residual blocks. Following these stages, a global average pooling layer aggregates each 8×8 feature map into a single scalar value, resulting in a 64-dimensional feature vector. Finally, a fully connected (FC) layer maps this vector to the desired number of output classes, producing the final classification logits. Notably, each residual layer group can employ a different quantization bit-width, allowing for layer-wise precision control that balances accuracy and efficiency. 3 layers consist of several residual blocks are quantized using equal, ascending, and descending bit precision.

Dataset	Model	Method	Test Accuracy	Train Accuracy	Train Loss	Train Time	Energy Consumption	Carbon Emission
CIFAR-10	ResNet18	Full Precision	90.01 ± 3.72	95.41 ± 5.36	0.1289 ± 0.1492	0.1562 ± 0.0377	$7.28 \times 10^{-6} \pm 4.27 \times 10^{-6}$	$2.40 \times 10^{-6} \pm 1.42 \times 10^{-6}$
		Ascending	86.88 ± 5.06	91.53 ± 0.24	0.2390 ± 0.1535	0.1525 ± 0.00093	$5.10 \times 10^{-6} \pm 2.42 \times 10^{-8}$	$1.69 \times 10^{-6} \pm 8.53 \times 10^{-9}$
		Descending	86.34 ± 5.63	90.23 ± 5.48	0.2766 ± 0.1514	0.1507 ± 0.00082	$5.10 \times 10^{-6} \pm 2.84 \times 10^{-8}$	$1.69 \times 10^{-6} \pm 9.38 \times 10^{-9}$
		Fixed Bit	84.26 ± 4.91	89.54 ± 5.48	0.2956 ± 0.1514	0.1529 ± 0.00061	$5.10 \times 10^{-6} \pm 5.48 \times 10^{-9}$	$1.69 \times 10^{-6} \pm 3.37 \times 10^{-9}$
CIFAR-100	ResNet18	Full Precision	64.36 ± 6.73	82.68 ± 10.69	0.56 ± 0.42	0.17 ± 0.01	$9.01 \times 10^{-6} \pm 3.62 \times 10^{-6}$	$2.97 \times 10^{-6} \pm 1.20 \times 10^{-6}$
		Ascending	55.69 ± 7.03	64.94 ± 7.87	1.21 ± 0.34	0.22 ± 0.03	$5.10 \times 10^{-6} \pm 2.88 \times 10^{-8}$	$1.68 \times 10^{-6} \pm 8.42 \times 10^{-8}$
		Descending	51.15 ± 6.36	61.8 ± 7.34	1.33 ± 0.32	0.146 ± 0.004	$5.11 \times 10^{-6} \pm 9.860 \times 10^{-8}$	$1.68 \times 10^{-6} \pm 1.22 \times 10^{-7}$
		Fixed Bit	54.75 ± 7.19	61.5 ± 7.37	1.35 ± 0.32	0.19 ± 0.04	$8.48 \times 10^{-6} \pm 3.66 \times 10^{-6}$	$2.73 \times 10^{-6} \pm 1.27 \times 10^{-6}$

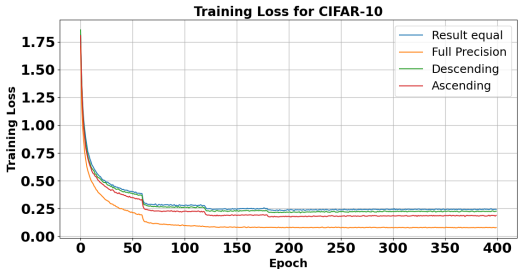
Table 3: Performance comparison of different methods on CIFAR-10 and CIFAR-100 datasets.

4. Experimental Setup

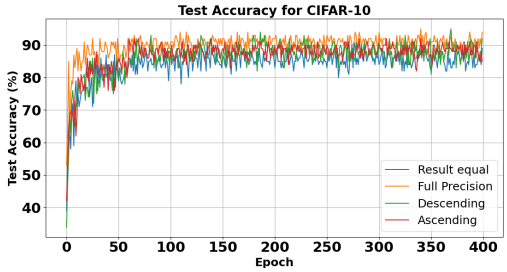
We conducted layer-wise quantization on the CIFAR-10 and CIFAR-100 [25] datasets, involving 400 epochs with a learning rate of 0.1. CIFAR-10 and CIFAR-100 each consist of 60,000 RGB images of size 32×32 pixels. However, CIFAR-10 is categorized into 10 classes, while CIFAR-100 includes a more fine-grained classification across 100 distinct classes. This configuration was aimed at evaluating how well a ResNet18 model performed while employing the layer-wise quantization technique, evaluating model efficiency, time, loss, and accuracy. The network was optimized using the SGD optimizer with a momentum of 0.9. We also conducted experiments using the full-precision model to analyze its energy consumption, enabling a comparative assessment of the efficiency gains achieved through



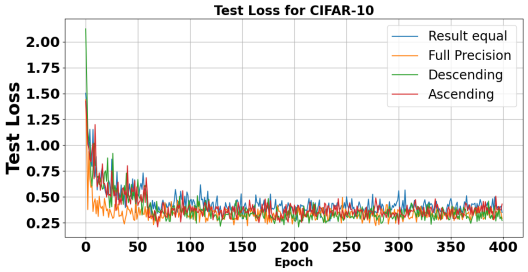
(a) Train Accuracy



(b) Train Loss

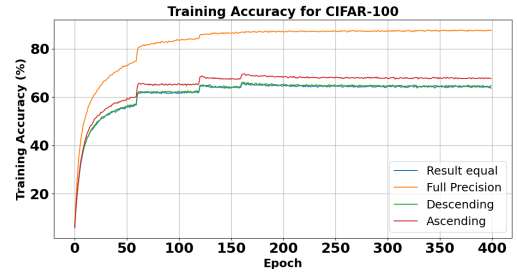


(c) Test Accuracy

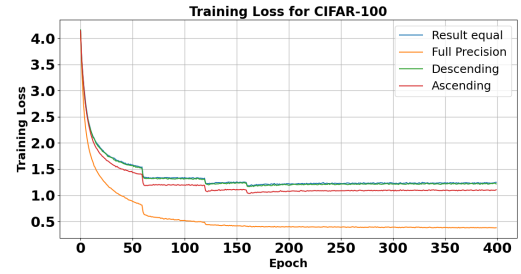


(d) Test Loss

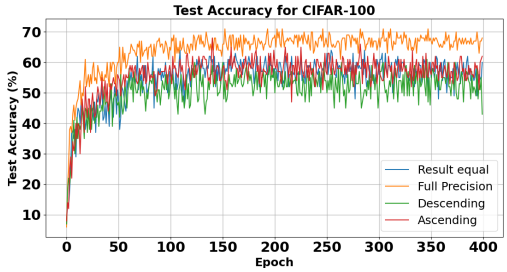
Figure 2: Accuracy and Loss with CIFAR-10



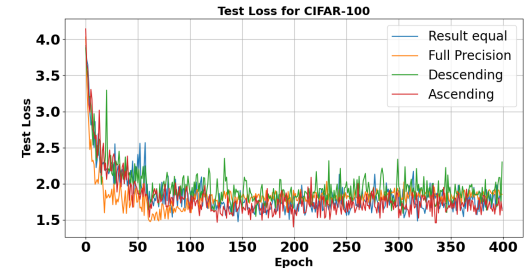
(a) Train Accuracy



(b) Train Loss



(c) Test Accuracy



(d) Test Loss

Figure 3: Accuracy and Loss with CIFAR-100

quantization. This quantization phase proved to be a key factor in decreasing the energy cost of the model's execution. We used the CodeCarbon software package [26] to accurately track the energy use and corresponding carbon emissions associated with the quantization technique. The experiments were performed using an Intel(R) Xeon(R) W-2265 CPU 3.50GHz, which has a Linux operating system of 76 cores and supports Metal 3, the latest graphics application programming interface (API) designed to boost graphics rendering on the platform.

4.1. Experimental Results

Table 3 presents a comparative analysis of different quantization strategies on CIFAR-10 and CIFAR-100 using ResNet18. Full precision achieves the highest accuracy but incurs the highest energy consumption and carbon emissions. Among quantized approaches, the ascending bit-width strategy consistently outperforms descending and fixed-bit configurations in both accuracy and loss, indicating that allocating higher precision to deeper layers improves performance. To ensure robustness, each experiment was conducted for 400 rounds. Performance is reported including both mean and standard deviation, as calculated using the statistical functions. These values reflect the variability observed within a single training run, based on the measurements collected throughout the training process. In particular, quantized methods significantly reduce energy and emissions, highlighting their potential for sustainable deployment, while maintaining competitive accuracy, especially on the less complex CIFAR-10 dataset. Figures 2 and 3 illustrate the training and testing accuracy and loss trends over communication rounds for CIFAR-10 and CIFAR-100, respectively. In CIFAR-10 (Figure 2), all quantization schemes converge stably, with full precision performing best, and ascending quantization closely following. Fixed-bit and descending strategies show slightly degraded accuracy and slower convergence. In CIFAR-100 (Figure 3), the performance gap widens, indicating greater sensitivity to quantization due to increased task complexity. Notably, ascending quantization again outperforms fixed and descending bit schemes, supporting the hypothesis that allocating higher precision to deeper layers preserves semantic fidelity and enhances learning stability.

Overall, the results demonstrate that although quantization techniques such as ascending, descending, and fixed-bit quantization may slightly reduce model accuracy, they provide significant advantages in energy efficiency, reduced communication overhead, and faster convergence. These benefits make them especially useful for neural network applications, where computational and resource efficiency are essential. These findings show the trade-off between accuracy and environmental impact, illustrating quantized training as a more energy-efficient alternative to deep learning, compared to full-precision training approaches. Although they are less accurate, these methods are far more energy efficient, and thus reasonable techniques to train resource-efficient models with low performance degradation.

5. Limitation

Apart from the benefits that layer-wise quantization offers, one of its major drawbacks is the loss of numeric accuracy, which can cause errors, especially with very low bit widths [18]. The errors introduced in each layer accumulate, and by the end of the process, the overall performance of the model is degraded [27]. This performance is significant in the case of deep networks. Moreover, due to the complexity of the network structure used and the presence of different types of data patterns, layer-wise quantization techniques may not work effectively in such scenarios, which in turn results in non-optimal quantization settings [11]. The optimal bit-width configuration for each layer in a model is often difficult and time consuming, especially for large models. A brute-force search is frequently impractical as the number of combinations is large [28]. Moreover, PTQ is typically faster than QAT, although some variations at the layer-wise level can be costly to compute. For instance, it is infeasible to search for the best bit-width of each layer using brute force search in large networks with larger number of layers [11]. Another problem in layer-wise quantization is the over-fitting of the training dataset by the model, which might hinder the ability of the trained model to perform well on new, unseen data. Furthermore, when considering different bit-widths per layer (mixed precision), the optimization problem is further exacerbated by an even larger search space [29]. Even though layer-wise quantization is a helpful technique for model compression, its possible drawbacks should also be considered, and a balance should be maintained among loss of accuracy, error propagation, and optimization level.

6. Discussion

In layer-wise quantization of ResNet18, different bit-widths are applied to different layers of the model to reduce communication overhead or memory usage, especially in on-device inference. When focusing on 3 specific layers, the choice of bit-width per layer can significantly impact model accuracy due to quantization error, communication cost, computational efficiency, and energy consumption. In our work, layer1, layer2, and layer3, each corresponding to groups of residual blocks and progressively deeper levels of feature extraction. Normally, the effect of equal quantization of 2, 2, and 2 bits in 3 different layers is uniform compression across all the layers. An extremely low bit width (2 bits) leads to high quantization error, especially in deeper or more critical layers that typically require more representational power. This may result in significant accuracy degradation, especially if the model is sensitive to weight precision. However, the advantage is the minimal communication, storage cost, and simplicity in implementation.

Layer	Feature Map Size	Channels	Functionality
Layer1	32×32	16	Low-level feature extraction
Layer2	16×16	32	Mid-level features, includes downsampling
Layer3	8×8	64	High-level abstraction, deep semantics

Table 4: Quantized layers in ResNet18 with their feature map sizes, channels, and functionalities.

According to Table 4, the features of layer 1 are edges and textures. Hence, layer 1 is less sensitive to quantization, especially at low bit-widths, and has a low risk of significant accuracy degradation [30]. Layer 2 gives combinations of low-level features. Hence, moderate sensitivity to quantization and important in preserving spatial coherence and abstraction [31]. Layer 3 features are abstract object parts that are critical for classification. These features are highly sensitive to quantization errors, and a low bit-width here can severely hurt model performance. In equal quantization, layer 3 suffers the most with reduced capacity to capture high-level semantics [32]. Therefore, equal quantization gives the best compression, minimal bandwidth/energy usage. Expected accuracy becomes significantly lower. However, equal quantization can be used in ultra-low-resource systems where accuracy is not the primary concern. In ascending quantization, lower bits for layer 1 are less critical, and higher bits for layer 3 are most critical. It balances compression and accuracy. In comparison to equal quantization, it slightly increased communication and computation cost. Its expected accuracy becomes closest to the full-precision baseline among the three. Also, it is more suitable for FL, where both efficiency and accuracy matter. Descending quantization prioritizes precision in layer 1 over layer 3. It preserves the fidelity of low-level features. However, the layer 3 quantization error leads to a loss in the quality of the final decision. Its accuracy is better than equal, worse than ascending. It is suitable for legacy systems using partial inference offloading, but not ideal in most cases.

7. Conclusion and Future Work

In this study, we analyzed the impact of layer-wise quantization on the performance of ResNet18 by targeting three core layers, each representing progressively deeper levels of feature abstraction. Through the application of equal, ascending, and descending quantization strategies with bit-widths of 2, 3, and 4, we demonstrated that the sensitivity to quantization is not uniform across layers. Specifically, lower layers (e.g., layer 1), which extract low-level features such as edges and textures, exhibit higher redundancy and tolerate aggressive quantization (e.g., 2-bit) with minimal accuracy degradation. In contrast, deeper layers like layer 3, responsible for high-level semantic representations, are significantly more sensitive to quantization errors. Our findings affirm that ascending quantization (e.g., 2, 3, 4 bits from shallow to deep layers) strikes the best trade-off between compression efficiency and model performance. The insights derived from this work highlight the importance of precision allocation in layer-wise quantization. By aligning quantization precision with the semantic importance of each layer,

one can optimize both resource utilization and inference accuracy, which is especially beneficial in resource-constrained environments such as edge devices and federated learning setups.

Future work will extend this study by exploring dynamic quantization schemes, where bit-widths are adjusted in real time based on layer-wise gradient statistics or activation distributions. Task-aware quantization, tailoring bit-widths according to the sensitivity of layers for specific downstream tasks (e.g., detection vs. classification). Energy-aware quantization frameworks that integrate energy profiling into the quantization decision process to further enhance the efficiency of on-device AI. Cross-layer optimization techniques, such as joint pruning and quantization, are used to holistically reduce computational overhead while maintaining performance. Adaptive training strategies that co-train bit-widths and weights to minimize quantization-induced degradation during training. By addressing these future directions, the community can move closer to achieving scalable, high-performance neural networks suitable for deployment in low-power, latency-sensitive applications.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

Acknowledgments

This work contributes to the basic research activities of the PNRR project FAIR - Future AI Research (PE00000013), Spoke 9 - Green-aware AI, under the NRRP MUR program funded by the NextGenerationEU.

References

[1] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), Advances in Neural Information Processing Systems, volume 25, Curran Associates, Inc., 2012.

[2] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805 (2018). arXiv:1810.04805.

[3] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, K. Zieba, End to end learning for self-driving cars, CoRR abs/1604.07316 (2016). arXiv:1604.07316.

[4] V. Sze, Y.-H. Chen, T.-J. Yang, J. S. Emer, Efficient processing of deep neural networks: A tutorial and survey, Proceedings of the IEEE 105 (2017) 2295–2329.

[5] R. Xu, W. Yu, Y. Liang, Towards efficient edge ai: A review on dnn quantization, IEEE Transactions on Neural Networks and Learning Systems 32 (2021) 4106–4120.

[6] A. Jobin, M. Ienca, E. Vayena, The global landscape of ai ethics guidelines, Nature Machine Intelligence 1 (2019) 389–399.

[7] R. Vinuesa, H. Azizpour, I. Leite, M. Balaam, V. Dignum, S. Domisch, A. Felländer, S. D. Langhans, M. Tegmark, F. Fuso Nerini, The role of artificial intelligence in achieving the sustainable development goals, Nature Communications 11 (2020) 1–10.

[8] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, Y. Bengio, Quantized neural networks: Training neural networks with low precision weights and activations, Journal of Machine Learning Research 18 (2017) 6869–6898.

[9] T. Qin, Z. Li, J. Zhao, Y. Yan, Y. Du, Mixed precision quantization based on information entropy, Scientific Reports 15 (2025) 12974.

[10] J. Jung, J. Kim, Y. Kim, C. Kim, Reinforcement learning-based layer-wise quantization for lightweight deep neural networks, in: 2020 IEEE International Conference on Image Processing (ICIP), IEEE, 2020, pp. 3070–3074.

[11] Y. Arai, Y. Ichikawa, Quantization error propagation: Revisiting layer-wise post-training quantization, arXiv preprint arXiv:2504.09629 (2025).
[12] R.-G. Dumitru, V. Yadav, R. Maheshwary, P. I. Clotan, S. T. Madhusudhan, M. Surdeanu, Layer-wise quantization: A pragmatic and effective method for quantizing LLMs, 2024. URL: https://openreview.net/forum?id=eJVrwDE086.
[13] S. Gluska, M. Grobman, Exploring neural networks quantization via layer-wise quantization analysis, arXiv preprint arXiv:2012.08420 (2020).
[14] H. Huang, H. Yu, Ltnn: A layerwise tensorized compression of multilayer neural network, IEEE transactions on neural networks and learning systems 30 (2018) 1497–1511.
[15] A. D. Nguyen, I. Markov, A. Ramezani-Kebrya, K. Antonakopoulos, D. Alistarh, V. Cevher, Layer-wise quantization for distributed variational inequalities, in: Workshop on Machine Learning and Compression, NeurIPS 2024, 2024.
[16] T. Shinde, Adaptive quantization and pruning of deep neural networks via layer importance estimation, in: Workshop on Machine Learning and Compression, NeurIPS 2024, 2024. URL: https://openreview.net/forum?id=kf6x9RCvHf.
[17] T. Shinde, R. Jain, A. K. Sharma, Lightweight neural networks for speech emotion recognition using layer-wise adaptive quantization, J. Name (2025).
[18] F. Zhang, Y. Liu, W. Li, J. Lv, X. Wang, Q. Bai, Towards superior quantization accuracy: A layer-sensitive approach, arXiv preprint arXiv:2503.06518 (2025).
[19] X. Zhao, R. Xu, Y. Gao, V. Verma, M. R. Stan, X. Guo, Edge-mpq: Layer-wise mixed-precision quantization with tightly integrated versatile inference units for edge computing, IEEE Transactions on Computers (2024).
[20] X. Zhu, W. Zhou, H. Li, Adaptive layerwise quantization for deep neural network compression, in: 2018 IEEE International Conference on Multimedia and Expo (ICME), IEEE, 2018, pp. 1–6.
[21] R.-G. Dumitru, V. Yadav, R. Maheshwary, P.-I. Clotan, S. T. Madhusudhan, M. Surdeanu, Layer-wise quantization: A pragmatic and effective method for quantizing llms beyond integer bit-levels, 2024. arXiv:2406.17415.
[22] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, T. Blankevoort, A white paper on neural network quantization, CoRR abs/2106.08295 (2021). arXiv:2106.08295.
[23] C. Yu, S. Yang, F. Zhang, H. Ma, A. Wang, E.-P. Li, Improving quantization-aware training of low-precision network via block replacement on full-precision counterpart, 2024. arXiv:2412.15846.
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 2016, pp. 770–778.
[25] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Technical Report, University of Toronto, 2009.
[26] B. Courty, V. Schmidt, S. Luccioni, Goyal-Kamal, MarionCoutarel, B. Feld, J. Lecourt, LiamConnell, A. Saboni, Inimaz, et al., mlco2/codecarbon: v2.4.1, 2024.
[27] Z. Xu, S. Sharify, W. Yazar, T. Webb, X. Wang, Understanding the difficulty of low-precision post-training quantization of large language models, arXiv e-prints (2024) arXiv–2410.
[28] D. Bablani, J. L. Mckinstry, S. K. Esser, R. Appuswamy, D. S. Modha, Efficient and effective methods for mixed precision neural network quantization for faster, energy-efficient inference, arXiv preprint arXiv:2301.13330 (2023).
[29] L. Wei, Z. Ma, C. Yang, Q. Yao, Advances in the neural network quantization: A comprehensive review, Applied Sciences 14 (2024) 7445.
[30] A. Zhou, A. Yao, Y. Guo, L. Xu, Y. Chen, Incremental network quantization: Towards lossless CNNs with low-precision weights, in: International Conference on Learning Representations, 2017.
[31] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, D. S. Modha, Learned step size quantization, in: International Conference on Learning Representations, 2020.
[32] R. Banner, Y. Nahshan, D. Soudry, Post training 4-bit quantization of convolutional networks for rapid-deployment, Curran Associates Inc., Red Hook, NY, USA, 2019.