COMPUTER SCIENCES

UDC 004.032.26 (045)
DOI:10.18372/1990-5548.87.20773

Dmytro Prochukhan

A METHOD FOR PREPARING CONVOLUTIONAL NEURAL NETWORKS FOR EDGE DEPLOYMENT

Kharkiv National University of Radio Electronics, Kharkiv, Ukraine
E-mail: viprochuhan@gmail.com ORCID 0000-0002-4622-1015

Abstract*—This study addresses the critical problem of accuracy loss during the compression of deep neural networks for mobile platforms. The research focuses on optimizing convolutional neural networks for operation under constrained hardware resources and the Memory Wall effect. An innovative Edge-deployment preparation method is proposed, which, unlike traditional sequential approaches, integrates structured pruning, post-training quantization, and a fine-tuning stage into a single iterative cycle. This approach provides a synergistic effect, minimizing accuracy degradation while achieving maximum parameter compression. Comparative analysis results confirm that the developed method meets strict latency and power consumption constraints, which are vital for mobile diagnostics in medical applications. Future research prospects involve adapting this method to other machine learning architectures.*

Keywords—Artificial intelligence; machine learning; neural networks; Edge deployment; pruning.

I. INTRODUCTION

Model compression is a mandatory requirement for deploying convolutional neural networks on Edge devices. The traditional approach involves the isolated or sequential application of methods such as pruning and quantization. This approach has significant drawbacks. The first issue is cumulative accuracy degradation. Applying pruning followed by quantization amplifies the errors from the first stage during the second, leading to substantial accuracy loss on Edge devices. The second issue is the sub-optimal utilization of Neural Processing Units (NPUs). Pruning without considering subsequent quantization leaves 'holes' in weight matrices that are difficult for NPUs to process efficiently. Conversely, quantization without prior pruning results in the quantization of numerous insignificant weights, wasting valuable computational resources and memory. The third issue is the lack of synergy. Both methods affect different aspects of the model and possess significant potential for synergistic interaction. An integrated approach allows a pruned network with fewer parameters to be less sensitive to quantization, while a quantized network can better withstand pruning if the feature space is robust. Consequently, there is a need for a method that integrates the benefits of both quantization and pruning simultaneously.

II. PROBLEM STATEMENT

Modern convolutional neural networks (CNNs) demonstrate high accuracy; however, migrating them from high-performance server data centers to resource-constrained devices encounters significant systemic hurdles. This transition requires a fundamental shift in design and implementation methodology, focusing on key engineering metrics: latency, power consumption, and hardware architectural efficiency. Deploying CNNs on mobile devices and Edge platforms necessitates a departure from universal frameworks in favor of lightweight and optimized execution environments [1]. The primary objective of such environments is to ensure efficient conversion of the trained model and maximum utilization of specialized hardware accelerators.

Specialized runtimes, such as TensorFlow Lite and Core ML, utilize an operation fusion mechanism. This process combines sequential elementary layers into a single, high-efficiency hardware block. This reduces overhead from repeated memory access between layers and increases overall arithmetic intensity, which is critical for performance [2], [3].

Ignoring hardware specifics is the most significant drawback of traditional CNN design methodologies, especially when porting server-side solutions to mobile platforms. On mobile chips, the primary performance bottleneck is not the volume of computations, but the energy and time spent on data movement – a phenomenon known as the Memory Wall. The term "Memory Wall" describes a situation where processor speed increases much faster than

memory access speed, causing system performance to be limited by slow memory latency.

According to research [3], the cost of accessing external DRAM can exceed the cost of a single 32-bit Multiply-Accumulate (MAC) operation by hundreds of times. Therefore, optimization efforts must primarily target minimizing external memory access and maximizing data reuse.

Quantization, which involves reducing the precision of weights and activations from FP32 to INT8, is the most effective method for ensuring energy savings and accelerating inference on resource-constrained devices. The transition to integer arithmetic provides energy efficiency through two interconnected mechanisms. Since INT8 data occupies four times less space than FP32, the shift to low precision proportionally reduces the volume of data transferred from DRAM to the chip by a factor of four. This effect directly counters the Memory Wall and provides substantial energy gains. Affine quantization is used to maintain the accuracy required for deep neural networks, linearly mapping the floating-point range to a limited integer range [4], [5]. Two main methodologies are used to apply quantization. They have different trade-offs between accuracy and engineering cost. Post-training quantization is the simplest and fastest method, as it is applied to an already trained model without additional training [6]. It can lead to uncontrolled degradation of accuracy when compressed aggressively.

This is unacceptable in medical image processing applications. [7]. Quantization-aware training is used to overcome this problem. Such quantization simulates exact INT8 arithmetic directly during training [7].

Advanced calibration methods are required to overcome calibration difficulties, such as Kullback–Leibler (KL) divergence-based calibration. This method minimizes information loss, ensuring the distribution of quantized values matches the original FP32 distribution as closely as possible while preserving vital entropy [8] – [12].

Pruning is another vital compression technique, but its engineering efficiency on hardware accelerators depends entirely on its structure. Unstructured pruning does not yield real-world speedups on standard CPUs or GPUs. For Edge deployment, structured pruning is critical, as it involves removing entire blocks, filters, or channels. This approach maintains the regularity of the remaining matrices, allowing hardware to efficiently apply SIMD (Single Instruction, Multiple Data) instructions.

The emergence of hardware-oriented methods like N:M sparsity [13] demonstrates that ideal pruning should impose a predictable, hardware-friendly zeroing pattern. Channel Pruning focuses on ranking and removing entire filters based on their importance, reducing both parameter count and computational cost [11] – [16].

Convolutional neural network optimization for Edge devices must be integrated into the architectural design process rather than being limited to post-training methods. The most progressive approach, aligned with computer engineering requirements, is Co-design.

This involves the simultaneous optimization of the neural network architecture and the hardware accelerator configuration. Since developing specialized accelerators is costly and time-consuming, Co-design aims to shorten this cycle by integrating hardware constraints into the training and Neural Architecture Search (NAS) process [17]. AutoML Co-design methods utilize sophisticated algorithms, such as Bayesian optimization, to jointly find the optimal "algorithm-accelerator" pair [18], [19].

Existing CNN optimization approaches are often isolated, leading to a substantial gap between potential and actual efficiency. Applying pruning without considering hardware architecture fails to provide a real reduction in latency, while applying quantization to unstable models leads to uncontrolled accuracy degradation.

The goal of this research is the development of an innovative method for preparing CNNs for further edge-based deployment.

III. Problem Solution

Studies [20] and [21] proposed a class-oriented augmentation method and a neural network integrating a Fourier layer. The application of these methods improves model accuracy and creates a margin of robustness during augmentation. However, deploying models integrated with these methods on Edge devices faces the challenge of accuracy loss. We shall develop an Edge-deployment preparation method that maintains accuracy under maximum compression. Preserving accuracy during maximum compression is a mandatory requirement for deployment on mobile phones.

The method treats pruning and quantization not as separate steps, but as interconnected parts of a single iterative process. This approach preserves the potential for Edge deployment. It is necessary to achieve an optimal accuracy-efficiency trade-off, as well as to adapt the models for mobile phones, creating architectures ideally suited for integer

and sparse computations on mobile neural processing units.

Below is the formalization of the Edge-deployment preparation method for mobile applications. The first step is initialization. The initial model M_0 – a pre-trained hybrid neural network architecture. The target performance metrics for the mobile platform are as follows:

$$\text{Latency}_{\max} < 50 \text{ ms}, \text{Size}_{\max} < 10 \text{ MB}, \Delta_{\text{Acc}} < 2\%. \quad (1)$$

The iterative cycle is performed until the target metrics are achieved. Structured pruning is carried out at each iteration. Let layer l have (C_l) filters $\{F_j^l\}_{j=1}^{C^l}$. The significance of the jth filter is defined as:

$$S_j^l = |F_j^l|_1. \quad (2)$$

Filters with the lowest significance scores are removed until the desired fraction of preserved filters S_j^l is reached. The updated model is obtained as follows:

$$\theta_{\text{pruned}}^{(t)} = \mathcal{P}\left(\theta^{(t)}; \{r_t^l\}\right) \quad (3)$$

where $\mathcal{P}$ is the structural pruning operator.

This step reduces computational complexity and ensures a regular tensor structure, which is optimal for the Neural Processing Unit. Following pruning, a short fine-tuning cycle is performed on the augmented dataset $\mathcal{D}$:

$$\theta_{\text{ft}}^{(t)} = \arg\min_{\theta \in \Theta(S^{(t)})} \mathcal{L}(\theta; \mathcal{D}) + \lambda \mathcal{R}(\theta), \quad (4)$$

where $\mathcal{L}$ is the loss function; $\mathcal{R}$ is the regularizer; and λ is the regularization coefficient. This stage restores the lost accuracy and stabilizes the model.

In the next step, post-training quantization is applied. The transformation of real-valued weights r into integer values q is performed according to the formula:

$$q = \text{clip}\left(\text{round}\left(\frac{r}{S}\right) + Z, q_{\min}, q_{\max}\right), \quad (5)$$

where S is the scale factor; Z is the zero-point; and for the INT8 format: $q_{\min} = -128$, $q_{\max} = 127$.

The recovery of real values is performed as follows:

$$\tilde{r} = S(q - Z). \quad (6)$$

Parameters S and Z are determined during the calibration process on a representative dataset.

The result is a quantized model $\theta_q^{(t)}$, ready for execution on a mobile neural processing unit (NPU).

In the next step, evaluation is performed and stopping criteria are defined.

After each iteration, accuracy and hardware metrics (size, latency, and energy consumption) are evaluated on the validation set:

$$\Delta_{\text{Acc}}^{(t)} = \frac{\text{Acc}(\theta_0) - \text{Acc}(\theta^{(t)} \cdot q)}{\text{Acc}(\theta_0)} \cdot 100\%,$$
$$\text{Size}^{(t)} = \text{Size}(\theta^{(t)} \cdot q), \quad (7)$$
$$\text{Latency}^{(t)} = \text{Latency}(\theta_q^{(t)}; \text{NPU}).$$

If conditions (1) are met, the iterative process terminates, and the resulting model $\theta^* = \theta_q^{(t)}$ is accepted as optimal. Our method provides a synergistic combination of structured pruning, quantization, and fine-tuning within a single iterative cycle. This allows for minimizing accuracy loss while meeting strict constraints on speed, size, and power consumption – critical parameters for mobile medical applications. We apply this method to the neural network model presented in studies (1) – (2). Figure 1 presents a comparative analysis of standard pruning and quantization versus the developed method with an intermediate fine-tuning stage.

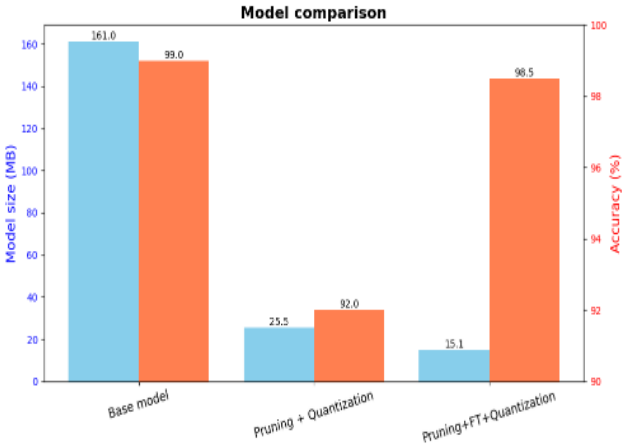

Fig. 1. Comparative analysis of compression results

IV. Conclusions

The developed method ensures the preservation of accuracy alongside maximum model compression. This is achieved through the joint application of pruning, quantization, and fine-tuning. The method enables the deployment of pre-trained neural network models on Edge devices. Future research prospects lie in applying this method to other machine learning tasks.

REFERENCES

[1] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90

[2] G Litjens. et al, "A survey on deep learning in medical image analysis", *Medical Image Analysis*, 2017, vol. 42, pp. 60–88. https://doi.org/10.1016/j.media.2017.07.005

[3] 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*, 2017, vol. 105, no. 12, pp. 2295–2329. https://doi.org/10.1109/JPROC.2017.2761740

[4] T. Elsken, J. H. Metzen, F. Hutter, "Neural Architecture Search: A Survey", *Journal of Machine Learning Research*, 2019, vol. 20, no. 55, pp. 1–21. https://doi.org/10.1007/978-3-030-05318-5_11

[5] L. Deng et al., "Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey", *Proceedings of the IEEE*, 2020, vol. 108, no. 4, pp. 485–532. https://doi.org/10.1109/JPROC.2020.2976475

[6] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv preprint arXiv:1704.04861, 2017.

[7] S. Han, H. Mao, W. J. "Dally Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding", *International Conference on Learning Representations (ICLR),* 2016.

[8] W. Wen et al., "Learning Structured Sparsity in Deep Neural Networks", *Advances in Neural Information Processing Systems* (NeurIPS), 2016, vol. 29.

[9] B. Jacob et al., "Quantization and Training of Neural Networks *ision and Pattern Recognition (CVPR)*, 2018, pp. 2704–2713. https://doi.org/10.1109/CVPR.2018.00286

[10] A. Gholami et al., "A Survey of Quantization Methods for Efficient Neural Network Inference", arXiv preprint arXiv:2103.13630, 2021. https://doi.org/10.1201/9781003162810-13

[11] M. S. Abdelfattah et al., "Best of Both Worlds: AutoML Codesign of a CNN and its Hardware Accelerator", *Proceedings of the 57th ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2020. https://doi.org/10.1109/DAC18072.2020.9218596

[12] C. Wu, "TFLite: Optimizing Mobile AI with Core ML Integration", Presentation Slides from Google IO, 2018.

[13] P. Micikevicius et al., "Mixed Precision Training for Deep Neural Networks", *International Conference on Learning Representations (ICLR)*, 2018.

[14] M. Sandler et al., "MobileNetV2: Inverted Residuals and Linear Bottlenecks", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2018, pp. 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[15] K. Hwang, J. H. Lee, "Survey of Hardware Accelerators for Deep Neural Networks", *IEEE Access*, 2018, vol. 6, pp. 48259–48280.

[16] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90

[17] G. Litjens et al., "A survey on deep learning in medical image analysis", *Medical Image Analysis*, 2017, vol. 42, pp. 60–88. https://doi.org/10.1016/j.media.2017.07.005

[18] 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*, 2017, vol. 105, no. 12, pp. 2295–2329. https://doi.org/10.1109/JPROC.2017.2761740

[19] T. Elsken, J. H. Metzen, F. Hutter, "Neural Architecture Search: A Survey", *Journal of Machine Learning Research,* 2019, vol. 20, no. 55, pp. 1–21. https://doi.org/10.1007/978-3-030-05318-5_11

[20] D. V. Prochukhan, "Fundux-oriented hybrid neural network with spatial-frequency processing and channel attention mechanism", *Information Processing Systems.* 2025, no. 3(182), pp. 70–75. https://doi.org/10.30748/soi.2025.182.07

[21] D. V. Prochukhan, "Class-oriented Method of Fundus Images Augmentation", *Visnyk of VPI*, no. 5, Oct. 2025, pp. 140–145. https://10.31649/1997-9266-2025-182-5-140-145

Received: November 24, 2025
Accepted: December 19, 2025
Published: February 11, 2026

Prochukhan Dmytro. ORCID 0000-0002-4622-1015. Postgraduate Student.
Kharkiv National University of Radio Electronics, Kharkiv, Ukraine.
Education: National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine, (2019).
Research interests: deep learning, artificial intelligence.
Publications: 63.
E-mail: viprochuhan@gmail.com

Д. В. Прочухан. Метод підготовки згорткових нейронних мереж до Edge-розгортання
У роботі вирішено актуальну проблему втрати точності при стисненні глибоких нейронних мереж для мобільних платформ. Об'єктом дослідження є процес оптимізації згорткових нейромереж для роботи в умовах обмежених апаратних ресурсів. Запропоновано інноваційний метод підготовки до Edge-розгортання, який, на відміну від традиційних послідовних підходів, інтегрує структуроване проріджування, пост-тренувальну квантизацію та етап донавчання в єдиний ітераційний цикл. Такий підхід забезпечує синергетичний ефект, дозволяючи мінімізувати деградацію точності при максимальному стисненні параметрів. Результати порівняльного аналізу підтверджують, що розроблений метод дозволяє досягти жорстких обмежень щодо латентності та енергоспоживання, що є критичним для мобільної діагностики у медичних застосунках. Перспективи подальших досліджень полягають в адаптації методу для інших архітектур машинного навчання.
Ключові слова: штучний інтелект; машинне навчання; нейронні мережі; Edge-розгортання; прунінг.

Прочухан Дмитро Володимирович. ORCID 0000-0002-4622-1015. Аспірант.
Харківський національний університет радіоелектроніки, Харків, Україна.
Освіта: Національний технічний університет «Харківський політехнічний інститут», (2019).
Напрям наукової діяльності: глибоке навчання, штучний інтелект, нейронні мережі.
Кількість публікацій: 63.
E-mail: viprochuhan@gmail.com