Electronics and Information Technologies, 2026, 35, 17–30
http://publications.lnu.edu.ua/collections/index.php/electronics/index

UDC: 004.89:004.93

HARDWARE-AWARE HYBRID COMPRESSION OF CONVOLUTIONAL NEURAL NETWORKS FOR RESOURCE-CONSTRAINED SYSTEMS

Andrian Kozynets* & *Lidia Demkiv*
System Design Department
Ivan Franko National University of Lviv
50, Drahomanova Street, Lviv, 79005, UKRAINE

Kozynets, A., Demkiv, L., (2026). Hardware-Aware Hybrid Compression of Convolutional Neural Networks for Resource-Constrained Systems. *Electronics and Information Technologies*, 35, 17–30. https://doi.org/10.30970/eli.35.2

ABSTRACT

Background. Deep neural networks face critical computational barriers for deployment on resource-constrained edge devices. While individual compression methods exist, their isolated application causes severe accuracy degradation. The synergistic combination of pruning, quantization, and knowledge distillation presents a solution, yet their interaction and hardware-specific efficiency remain insufficiently explored.

Materials and Methods. This study investigates hybrid compression pipelines applied to linear (VGG) and residual (ResNet) network architectures. We evaluated a Two-Stage Distillation algorithm featuring a preliminary classifier stabilization phase to eliminate gradient shock, combined with structural filter removal and static integer quantization. A cross-platform analysis was conducted on central and graphics processing units, and Pareto efficiency frontiers were constructed to determine the optimal trade-offs.

Results and Discussion. The preliminary stabilization phase successfully mitigated gradient shock, consistently improving student accuracy by up to 2.6%. While isolated structural pruning significantly degraded performance, distillation healing via teacher-generated soft targets enabled a pruned network with nearly 30% of weights removed to outperform its full-sized baseline. On CPUs, the hybrid pipeline reduced memory footprint by 11 times and accelerated inference by 5 times. Conversely, integer quantization demonstrated critical inefficiency on GPUs due to memory latency; for such accelerators, half-precision format was Pareto-optimal for massive linear architectures, whereas standard precision guided by two-stage distillation provided the best efficiency for residual models.

Conclusion. Integrating Two-Stage Distillation with structural pruning and quantization fundamentally outperforms isolated compression techniques. The optimal strategy is strictly hardware- and architecture-dependent: integer-based hybrid pipelines are essential for energy-efficient edge CPUs, while for GPU accelerators, the choice between half-precision and standard precision depends on the network's topological characteristics.

Keywords: Artificial intelligence, Deep learning, Neural network optimization, Resource-constrained systems, Hardware acceleration.

INTRODUCTION

The rapid evolution of artificial intelligence, particularly in the field of computer vision, has been largely driven by the increasing depth and complexity of convolutional neural networks (CNNs). Modern retrospective analyses of deep learning architectures demonstrate that foundational topologies, such as the ResNet and VGG families, achieve unprecedented benchmarks in image classification and object detection by utilizing

© 2026 Andrian Kozynets & Lidia Demkiv. Published by the Ivan Franko National University of Lviv on behalf of Електроніка та інформаційні технології / Electronics and Information Technologies. This is an Open Access article distributed under the terms of the Creative Commons Attribution 4.0 License which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

massive parametric spaces to extract highly complex, hierarchical feature representations [1]. However, this architectural scaling comes at a severe computational cost. Modern CNNs require gigabytes of memory for weight storage and billions of floating-point operations (FLOPs) per inference. Consequently, the deployment of state-of-the-art CNNs on edge devices – such as mobile platforms, autonomous drones, and Internet of Things (IoT) sensors – is strictly hindered by limited memory bandwidth, restricted thermal envelopes, and stringent energy budgets [2].

To bridge the substantial gap between high-performance deep learning models and the physical hardware constraints of peripheral devices, the scientific community has intensely focused on model compression techniques. Among these, Knowledge Distillation (KD) has emerged as a fundamental and highly effective approach. KD involves training a compact "student" model to replicate the functional behavior of a cumbersome "teacher" model by learning from the teacher's temperature-scaled probabilities (soft targets), which contain crucial dark knowledge regarding inter-class similarities [3]. While KD provides an elegant theoretical framework for information transfer, its isolated application often fails to achieve the extreme compression ratios required for real-time edge inference. Furthermore, recent statistical evaluations of distillation mechanisms indicate that applying KD directly to new, uninitialized domains frequently induces a "gradient shock," where chaotic initial gradients irreversibly disrupt the spatial feature hierarchies of the student model, significantly limiting knowledge absorption [4].

To achieve aggressive model minimization, structural pruning and quantization are actively employed. Structural pruning aims to reduce the model's physical width and computational complexity by completely removing redundant structural components, such as entire convolutional filters or feature channels. Unlike unstructured pruning, structural reduction directly translates to hardware acceleration without the need for specialized sparse matrix libraries [5]. Simultaneously, model quantization converts high-precision 32-bit floating-point (FP32) weights and activations into low-bitwidth integer formats, such as 8-bit integers (INT8) [6]. This allows models to leverage rapid vector instructions on modern Central Processing Units (CPUs), drastically reducing memory footprint and accelerating inference speed.

Despite the individual promise of these techniques, their practical implementation in hybrid pipelines faces profound algorithmic and hardware-related challenges. The simultaneous or sequential application of structural pruning and INT8 quantization radically alters the geometry of the parameter space. Standard fine-tuning using the conventional cross-entropy loss function is often incapable of navigating this severely restricted space, leading to a catastrophic and non-linear degradation in classification accuracy [7]. Furthermore, the efficiency of these compression pipelines is strictly hardware-dependent. A compression configuration optimized for a cloud-based Graphics Processing Unit (GPU) – which favors floating-point operations and massive parallelism – might result in severe memory bottlenecks when deployed on an edge CPU, and vice versa [2].

Therefore, there is an objective need for a formalized, hardware-aware optimization framework that synergizes these distinct compression paradigms. This study investigates the integration of a Two-Stage Distillation algorithm with hybrid compression pipelines to mitigate accuracy degradation. In particular, the phenomenon of “distillation healing” is explored, under the premise that teacher-generated soft targets can act as a precise spatial navigator for the optimizer, thereby recovering the representative capacity of pruned networks within a narrowed search space.

The main objective of this study is to develop, theoretically justify, and experimentally evaluate a hybrid compression framework that integrates Two-Stage Distillation, structural pruning, and static quantization. By conducting an extensive cross-platform analysis on CPU and GPU architectures, this research aims to determine Pareto-efficient configurations that provide an optimal balance between classification accuracy, memory

footprint, and real-time inference speed for resource-constrained artificial intelligence models.

MATERIALS AND METHODS

Neural Network Architectures and Experimental Framework. To ensure a comprehensive and architecture-agnostic evaluation of the proposed hybrid compression pipelines, the experiments were conducted using two fundamentally distinct convolutional neural network topologies. Although lightweight architectures such as MobileNet, EfficientNet, and RepVGG are widely adopted for edge deployment, this study intentionally focuses on the VGG and ResNet families. Recent studies continue to investigate compression and deployment techniques based on ResNet architectures for resource-constrained edge systems, demonstrating their ongoing relevance as benchmark backbones for evaluating optimization methods [8]. Furthermore, recent surveys also emphasize the coexistence of lightweight and conventional CNN architectures in modern edge AI research, where MobileNet, EfficientNet, RepVGG, VGG, and ResNet serve complementary roles depending on application requirements [9]. ResNet remains one of the most frequently used backbone architectures in modern computer vision applications, whereas VGG continues to serve as a widely accepted benchmark model in neural network compression research. Furthermore, these architectures represent two fundamentally different CNN design paradigms: sequential and residual, which enable the evaluation of the proposed compression framework across distinct network topologies. The first family is represented by the VGG architecture (specifically, VGG16 as the teacher and VGG11 as the baseline student). VGG networks are characterized by a deep, linear topology with massive fully connected layers, making them highly susceptible to gradient degradation and memory bottlenecks. The second family includes Residual Networks (ResNet50 as the teacher and ResNet18 as the student). Unlike VGG, ResNet incorporates skip connections (residual blocks) that act as safe gradient highways, providing inherent structural robustness and efficient feature propagation. Recent retrospective analyses confirm that evaluating compression techniques across both linear and residual topologies is mandatory to objectively determine a method's universal applicability [10].

Open-Source Models and Dataset Adaptation. To ensure the reproducibility and transparency of the research, all investigated models (VGG16, VGG11, ResNet50, and ResNet18) are standard, classical architectures. Their implementations and baseline weights were obtained from the publicly available open-source PyTorch repository (*torchvision.models*).

It is important to emphasize that the proposed hybrid compression methodology is fundamentally dataset-agnostic. However, to provide an objective and reproducible experimental baseline, all models were evaluated using the publicly accessible ImageNet100 dataset, a standardized benchmark containing 100 image classes.

Furthermore, the experimental protocol did not involve training the baseline models from scratch. Instead, the pre-trained architectures underwent a preliminary fine-tuning (adaptation) phase on the ImageNet100 dataset. This step was mandatory to align the feature space with the target domain before any destructive compression manipulations (such as structural pruning or INT8 quantization) and the subsequent distillation healing processes were applied. This approach strictly isolates the performance impact of the compression algorithms from the variances of initial training.

Implementation of Compression Algorithms. The core methodology of this study revolves around the synergistic application of destructive compression techniques and knowledge-guided recovery. The following discrete optimization algorithms were implemented:

Structural Filter Pruning. Unlike unstructured pruning, which zeroes out individual weights and requires specialized sparse-matrix hardware to realize speedups, we

implemented structural filter pruning based on the L_1-norm criterion [11]. In this approach, the absolute sum of the weights within each convolutional filter is calculated. Filters with the lowest L_1-norm are considered to contribute the least to the feature extraction process and are physically removed from the network architecture. This reduces the number of output channels, subsequently decreasing the number of multiply-accumulate (MAC) operations. In our experimental setup, a baseline sparsity level of 15% to 28% was targeted for the student models [5].

Static Integer Quantization (INT8). To maximize inference throughput on edge devices lacking powerful floating-point units, post-training static quantization (PTSQ) was utilized [12]. This process maps the 32-bit floating-point (FP32) weights and activations into an 8-bit integer (INT8) space. Because dynamic quantization calculates activation ranges during runtime (adding computational overhead), static quantization was chosen. It requires a preliminary calibration phase, where a representative subset of the validation dataset is passed through the network to compute the statistical distribution (minimum and maximum clipping ranges) of the activations.

Half-Precision Formatting (FP16). To leverage the specific hardware accelerators available on modern Graphics Processing Units (GPUs), a conversion to half-precision floating-point format (FP16) was implemented [13]. This reduces the memory bandwidth requirements by half compared to FP32, accelerating memory-bound operations without the severe precision loss associated with aggressive integer quantization.

Knowledge Distillation as a Healing Mechanism. While the mathematical formulation and initialization stabilization of the Two-Stage Distillation algorithm are outside the scope of this paper, its role in this research acts exclusively as a recovery protocol. Instead of fine-tuning the severely damaged pruned architectures using standard hard-label cross-entropy, the temperature-smoothed soft targets ($T = 4.0$) generated by the robust teacher model are used to guide the student's optimizer. This "distillation healing" provides dense spatial navigation through the narrowed parameter space [3].

Classification of Investigated Compression Pipelines. To systematically isolate the effects of individual and hybrid compression techniques, a rigid matrix of experimental pipelines was established. **Table 1** defines the ten distinct methodologies evaluated in this study.

Table 1. **List of experimental optimization methods**

No	Method name
00	Baseline (Teacher)
01	Quantization Static (INT8)
02	Pruning Structured(15%)
03	Self-Distillation
04	Distillation + Quantization
05	Baseline Student
06	Two-Stage Distillation (Student)
07	Distillation + FP16
08	Distillation + Prune
09	Distillation + Prune + Quantization

Baselines (Methods 00 & 05). Method 00 (Teacher) serves as the theoretical upper bound for accuracy and the source of "dark knowledge" (ResNet50 or VGG16). Method 05

(Baseline Student) represents the lightweight architecture trained from scratch using standard cross-entropy, illustrating the architectural capacity gap.

Isolated Classical Compression (Methods 01 & 02). These pipelines apply compression without distillation guidance. Method 01 involves isolated static INT8 quantization. Method 02 physically removes 15% of convolutional filters using L_1-norm pruning, followed by standard hard-label fine-tuning.

Standard Distillation Indicators (Methods 03 & 04). Method 03 represents the classic self-distillation protocol, forcing a randomly initialized student to match the teacher's KL-divergence immediately. Method 04 follows this with standard quantization. These methods serve to demonstrate the vulnerability of networks to unmitigated gradient shocks.

Advanced Distillation and Healing (Methods 06, 07, & 08). Method 06 is the isolated application of the Two-Stage Distillation protocol. Method 07 converts the distilled student (Method 06) into FP16 format, designed specifically for GPU tensor core acceleration. Method 08 is the core "distillation healing" pipeline, where the student model undergoes aggressive structural pruning and is subsequently fine-tuned using the teacher's soft targets.

Extreme Hybrid Compression (Method 09). This is the most aggressive pipeline, engineered for severely constrained edge environments. It sequentially combines distillation (for maximal feature retention), structural pruning (for topological reduction), and static INT8 quantization (for vector-instruction acceleration).

Hardware Platforms and Cross-Platform Testing Environment. The computational efficiency of any model compression pipeline is fundamentally hardware-dependent. To comprehensively evaluate the proposed hybrid methods, empirical testing was conducted across two distinct hardware environments, representing the two primary deployment scenarios: resource-constrained edge computing and high-performance cloud inference [2].

The first environment, serving as a proxy for edge devices, utilized an AMD Ryzen 5 5600 multi-core Central Processing Unit (CPU) supporting Advanced Vector Extensions (SIMD instructions) and 16 GB of system memory. This platform was explicitly chosen to evaluate the hardware acceleration capabilities of static INT8 quantization (Methods 01, 04, and 09), as integer arithmetic allows the CPU to process vectorized data efficiently, significantly reducing latency [12]. The second environment utilized an NVIDIA GeForce RTX 3060 Ti Graphics Processing Unit (GPU) equipped with Tensor Cores and Compute Unified Device Architecture (CUDA) support. This environment was used to measure baseline models and to test the specific efficacy of half-precision (FP16) formatting (Method 07) [13].

Also, all performance measurements reported in this study were obtained using a batch size of 32.

Evaluation Metrics and Data Processing. To quantitatively assess the effectiveness of the distillation and hybrid compression algorithms, three primary metrics were recorded for each experimental pipeline:

1. Classification Accuracy. Measured as the standard validation accuracy on the testing dataset. This metric serves as the primary indicator of the feature space integrity and the success of the distillation healing mechanism.

2. Memory Footprint and Compression Ratio. The physical storage size of the serialized model weights, measured in Megabytes (MB). The Compression Ratio is calculated as the baseline teacher model size divided by the compressed student model size. This metric is crucial for determining the feasibility of deployment on memory-constrained peripheral devices.

3. Inference Speed (Throughput). Measured in Frames Per Second (FPS). To isolate the computational throughput of the network from data-loading bottlenecks, FPS was calculated based on model forward-pass execution times, allowing for an objective cross-platform speed comparison.

Since no single pipeline simultaneously maximizes accuracy and minimizes latency across all hardware, a Pareto-efficiency analysis was employed. Constructing Pareto

frontiers allows for an objective mathematical comparison of the trade-offs inherent to different compression strategies. All experimental data, including neural network weights, validation accuracies, and inference latency measurements, were collected and processed natively within the Python programming environment using the PyTorch deep learning framework [14].

RESULTS AND DISCUSSION

The "Distillation Healing" Phenomenon: Restoring Capacity After Pruning. A key objective of the experimental phase was to evaluate the capacity of knowledge distillation to preserve feature space integrity during structural pruning. Structural pruning physically removes convolutional filters based on their L_1-norm, radically altering the geometry of the parameter space and permanently reducing the model's capacity [11]. The isolated application of this method, followed by standard fine-tuning using hard categorical labels, typically fails to allow the model to adequately recover its lost representational pathways. This occurs because the narrowed architecture lacks the dense parameterization required to navigate the complex loss landscape from scratch. To evaluate this, we compared isolated pruning against our hybrid approach, wherein the pruned student architecture is trained using the temperature-smoothed "soft targets" generated by the uncompressed teacher model.

The empirical data presented in **Table 2** substantiates the severe impact of unguided compression on the ResNet architecture. The application of standard structural pruning (Method 02) demonstrably degrades the network's accuracy, dropping it to 0.839 relative to the baseline teacher.

Table 2. **The impact of distillation on robustness to structural pruning (ResNet family).**

Method №	Accuracy	Sparsity
01 (Quantization Static (INT8))	0.845	0.020
02 (Pruning Structured(15%))	0.839	0.121
04 (Distillation + Quantization)	0.851	0.020
08 (Distillation + Prune)	0.778	0.287
09 (Distillation + Prune + Quantization)	0.789	0.296

However, the "distillation healing" phenomenon is prominently manifested in Method 08. Following the aggressive removal of approximately 28% of the convolutional weights in the ResNet18 student model, the introduction of teacher-guided soft targets enabled the highly sparse model to achieve an accuracy of 0.778. This results in a highly counterintuitive paradox: the pruned, distilled student model strictly outperforms the fully dense baseline student model trained from scratch (Method 05, which only achieved an accuracy of 0.770).

This outcome provides strong empirical evidence that knowledge distillation does not merely act as an initialization strategy but serves as an ideal spatial navigator. The soft targets contain essential "dark knowledge" about inter-class similarities, which effectively guides the optimizer to find highly efficient signal routing paths within the strictly constrained parameter space of the pruned network [3].

As illustrated in **Table 3**, an analogous "healing" effect is observed for linear topologies, which are traditionally much more vulnerable to structural damage than residual networks due to the lack of gradient bypass connections [15]. Standard pruning applied to the VGG architecture (Method 02) causes a steep collapse in accuracy down to 0.635.

Table 3. **The impact of distillation on robustness to structural pruning (VGG family).**

Method №	Accuracy	Sparsity
01 (Quantization Static (INT8))	0.649	0.017
02 (Pruning Structured(15%))	0.635	0.016
04 (Distillation + Quantization)	0.623	0.017
08 (Distillation + Prune)	0.64	0.498
09 (Distillation + Prune + Quantization)	0.635	0.479

Nevertheless, the application of the hybrid pipeline (Method 08) successfully mitigates this degradation. By fine-tuning the pruned VGG network with distillation gradients, the feature space is partially restored, allowing the network to reach an accuracy of 0.640.

Remarkably, just as observed with the ResNet family, this heavily pruned VGG network outperforms the full-sized baseline VGG11 student (Method 05, accuracy 0.616). This confirms that the distillation healing mechanism is architecture-agnostic and universally beneficial for restoring the capacity of sparse convolutional topologies.

Deep Compression and Memory Footprint Reduction. For the successful deployment of convolutional neural networks on peripheral edge devices (such as IoT sensors or mobile processors), the physical memory footprint of the serialized model is as critical as its classification accuracy. Extreme reduction of the parameter space is required to fit these models into limited on-chip memory and to reduce energy-consuming off-chip memory accesses [2].

Fig. 1 illustrates the compression trade-off landscape for the ResNet family. The most impressive result of extreme memory reduction is demonstrated by the aggressive hybrid pipeline, Method 09. The sequential application of Two-Stage Distillation, structural pruning, and static INT8 quantization reduces the physical memory footprint from 84.2 MB (baseline teacher, Method 00) to an unprecedented 7.7 MB. This achieves a massive compression ratio of 10.9x. Despite this drastic reduction in capacity, the preserved accuracy stands at 0.789, strictly outperforming all other standalone compact model configurations represented on the chart.

As observed in **Fig. 2**, the massive VGG architecture exhibits an even more pronounced response to hybrid compression. Method 09 successfully reduces the VGG memory footprint from 513.7 MB (baseline teacher) to a mere 74.4 MB. This represents a compression ratio of 6.9x, successfully eliminating the vast majority of redundant parameters stored in the dense fully connected layers.

Throughout this aggressive reduction, the model maintains a highly competitive accuracy of 0.635. It is crucial to note that attempts to achieve similar compression ratios without the proposed Two-Stage Distillation mechanism – such as Method 04, which utilizes standard distillation alongside quantization – yield significantly worse results, dropping accuracy to 0.623. This further validates that advanced distillation is mandatory to preserve the feature space during aggressive 8-bit quantization [6].

Pareto-Efficiency and Cross-Platform Hardware Analysis. The hardware architecture of the target computing device strictly dictates the efficacy of any given compression methodology. A pipeline that performs exceptionally well in a cloud environment may fail entirely on an edge device. To objectively evaluate these trade-offs, Pareto efficiency frontiers were constructed. A compression configuration is considered Pareto-optimal if no other tested method can provide a higher inference speed (Frames Per Second, FPS) without a corresponding degradation in classification accuracy.

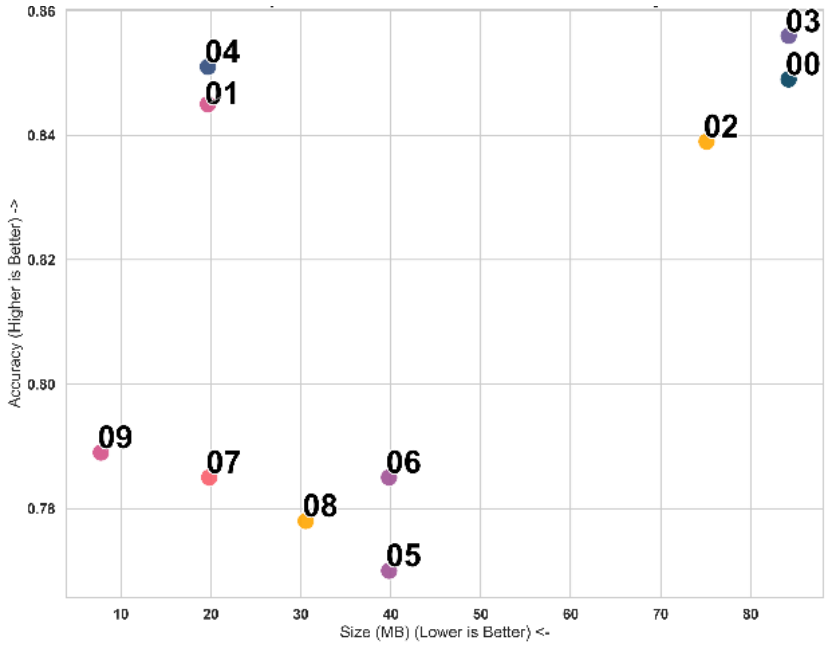


Fig. 1. Trade-off between model size and classification accuracy for the ResNet architecture

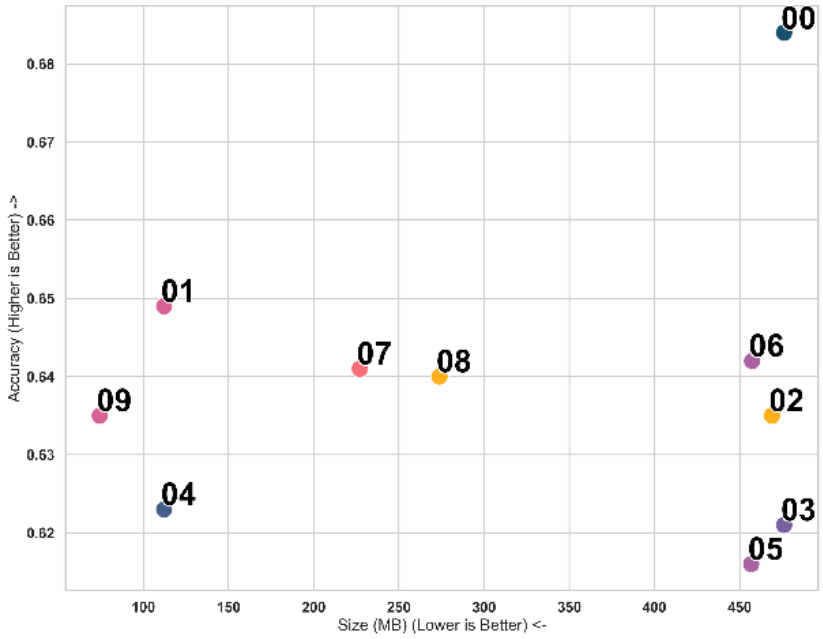


Fig. 2. Trade-off between model size and classification accuracy for the VGG architecture

Fig. 3 visualizes the performance of the VGG architectural family on a Central Processing Unit (CPU), simulating a resource-constrained edge computing environment. The massive, linear structure of VGG traditionally suffers from severe latency on CPUs. The Pareto frontier clearly indicates that Method 09 – the aggressive hybrid pipeline combining Two-Stage Distillation, structural pruning, and static INT8 quantization – is the absolute optimal solution. By translating the 32-bit floating-point weights into 8-bit integers,

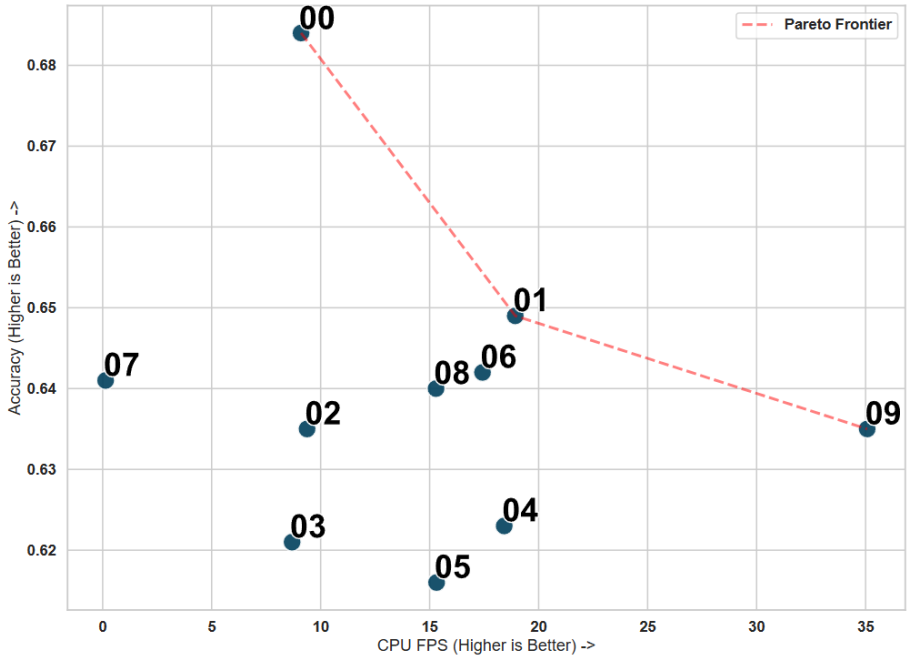


Fig. 3. Pareto efficiency frontier of compression methods for VGG on a CPU platform

the CPU can maximally utilize Advanced Vector Extensions (SIMD instructions) to process multiple operations simultaneously [12].

Consequently, the inference speed increases dramatically from a baseline of 9.09 FPS (Method 00, Teacher) to 35.07 FPS, representing a 3.85x acceleration while maintaining a competitive accuracy of 0.635. Furthermore, the graph demonstrates that isolated static quantization (Method 01) yields only 18.9 FPS, proving that integer arithmetic alone is insufficient for heavy architectures without the topological reduction provided by structural pruning.

The dynamics shift radically when deploying the VGG network on a platform equipped with Compute Unified Device Architecture (CUDA) and specialized Tensor Cores, as illustrated in **Fig. 4**. Notably, all pipelines utilizing INT8 quantization (Methods 01, 04, and 09) are entirely absent from the Pareto frontier. In hardware-accelerated environments, these integer-based methods demonstrate a critical performance collapse, recording 0.0 effective FPS. It is important to emphasize that this observation does not reflect a hardware limitation of modern GPUs but rather the characteristics of the native software environment used in this study. All experiments were conducted exclusively within the standard PyTorch framework without employing specialized inference deployment or optimization frameworks. Under these conditions, PyTorch does not provide fully optimized native execution kernels for direct INT8 inference on GPUs. Within the native PyTorch execution pipeline used in this study, software dequantization and associated memory bandwidth overheads prevent efficient utilization of dedicated GPU integer execution units, resulting in severe performance degradation [16].

Instead, Method 07 dictates the Pareto front. Achieving efficient INT8 inference on modern GPUs typically requires deploying the model through specialized optimization frameworks, such as NVIDIA TensorRT, AMD MIGraphX, or Intel OpenVINO, rather than relying solely on the native PyTorch execution environment used in this work. By converting the distilled student model into a half-precision floating-point format (FP16), the network perfectly aligns with the GPU's native Tensor Core capabilities. This allows the VGG network to be processed at an exceptional 601.78 FPS, outperforming the original teacher

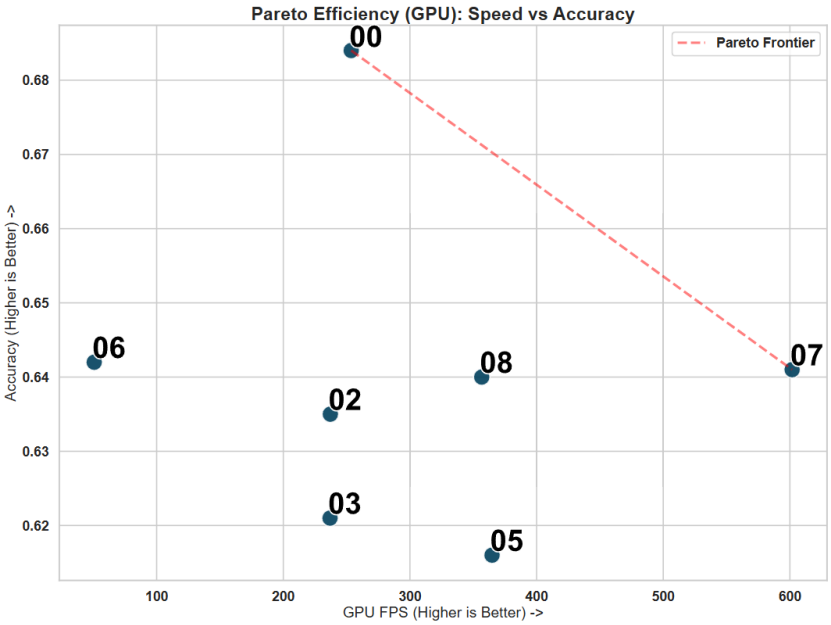


Fig. 4. Pareto efficiency frontier of compression methods for VGG on a GPU platform

(253.6 FPS) by 2.37 times, with minimal accuracy fluctuation (0.641). **Fig. 5** examines the CPU performance of the ResNet family. Inherently more efficient than VGG due to residual connections and the absence of massive fully connected layers, the baseline ResNet50 (Method 00) processes at 23.6 FPS.

However, the Pareto frontier reveals that the Two-Stage Distillation algorithm combined with hybrid compression (Method 09) maximizes hardware efficiency far beyond the baseline. Just as observed with the VGG network, Method 09 forms the optimal peak

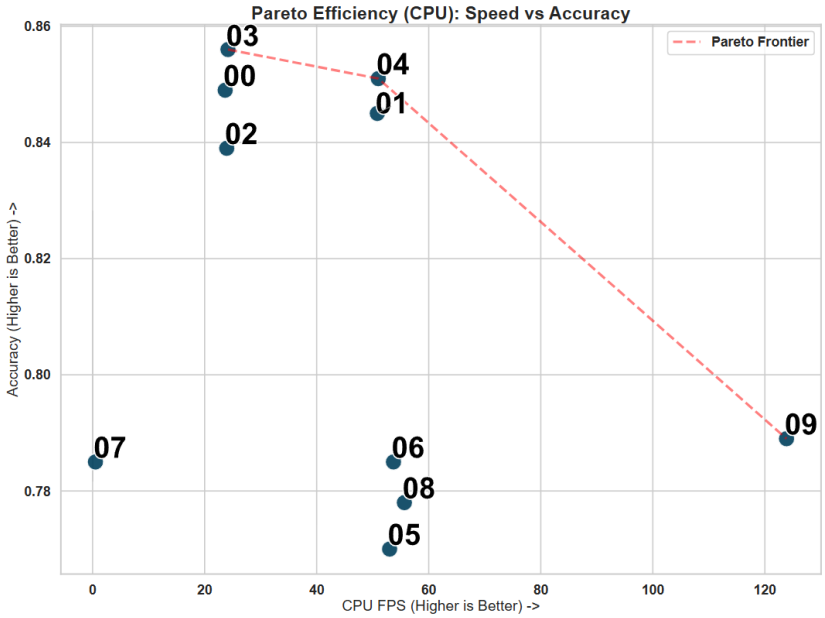


Fig. 5. Pareto efficiency frontier of compression methods for ResNet on a CPU platform

on the CPU. The inference speed leaps from 23.6 FPS to a phenomenal 123.8 FPS, achieving a 5.24x acceleration. Because the accuracy is preserved at 0.789, this specific pipeline proves to be the ideal candidate for deploying real-time, high-accuracy computer vision systems on embedded IoT processors.

Finally, **Fig. 6** demonstrates the performance of the compressed ResNet models on a high-performance GPU. Mirroring the VGG GPU analysis, the application of integer quantization (INT8) leads to zero effective throughput within the native PyTorch environment, completely excluding Methods 01, 04, and 09 from the Pareto frontier. As discussed above, this behavior reflects the limitations of the native software execution environment rather than the hardware capabilities of modern GPUs.

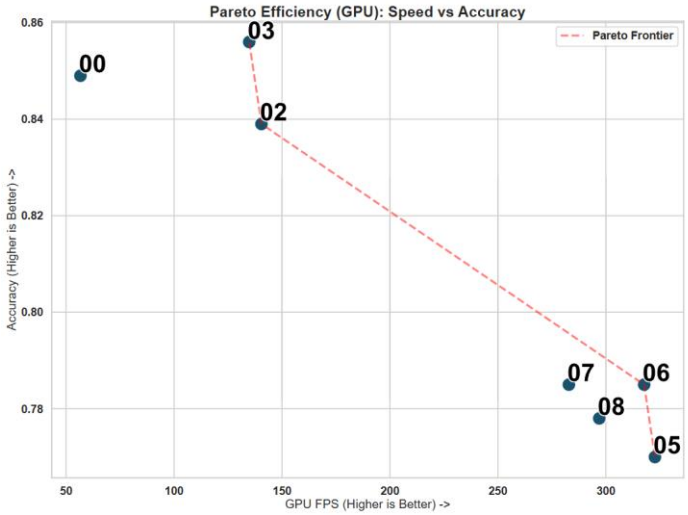


Fig. 6. Pareto efficiency frontier of compression methods for ResNet on a GPU platform

However, the dynamic for ResNet diverges significantly from the VGG family regarding half-precision formatting. On this CUDA-supported platform, Method 07 (FP16) does *not* lie on the Pareto efficiency frontier. Instead, the optimal frontier is formed by pure distillation-based models operating in standard precision: Method 03, Method 02, Method 06, and Method 05. The Two-Stage Distillation applied to the student model in standard FP32 precision (Method 06) achieves a superior balance, reaching an impressive inference speed of approximately 315 FPS with an accuracy of 0.785. In contrast, Method 07 achieves the exact same accuracy but lags in speed (processing at approximately 280 FPS).

This indicates that for inherently optimized, compact architectures like ResNet18, standard 32-bit execution (Method 06) outperforms FP16 conversion, avoiding the hardware overhead associated with precision casting [13]. Method 06 emerges as the definitive Pareto-efficient choice for achieving extremely high throughput on GPUs without compromising the integrity of the distilled feature space.

CONCLUSION

In this research, we have successfully addressed the scientific and practical challenge of enhancing the efficiency of knowledge transfer between convolutional neural networks for deployment on resource-constrained edge devices. Through comprehensive theoretical analysis and extensive experimental modeling across multiple hardware backends, several critical conclusions have been established.

First, the study confirms that standard knowledge distillation protocols are highly vulnerable to the phenomenon of "gradient shock" during the initialization phase,

particularly when applied to linear topologies. For the VGG family, the absence of a stabilization phase resulted in a 6.3% drop in classification accuracy. The implemented Two-Stage Distillation algorithm effectively mitigates this issue by introducing a preliminary stabilization phase. By temporarily freezing the deep convolutional layers and training only the newly initialized classifier, this "warmup" safely isolates and absorbs destructive gradients. This stabilization process consistently improved the accuracy of compact student models, yielding a gain of +2.6% for VGG11 and +1.5% for ResNet18 compared to baseline training from scratch.

Second, the experimental results provide robust evidence for the existence of the "distillation healing" mechanism. We demonstrated that utilizing temperature-smoothed soft targets from a teacher model acts as a powerful spatial navigator for the student's optimizer within a narrowed parameter space. This recovery mechanism allowed structurally pruned architectures to not only mitigate the damage from the removal of nearly 30% of their weights but also to outperform their full-sized baseline student counterparts in accuracy.

Third, the cross-platform analysis established that the optimal compression strategy is strictly hardware-dependent. For edge computing systems (CPU), a hybrid pipeline combining Two-Stage Distillation, structural pruning, and static INT8 quantization emerged as the only viable solution for extreme efficiency. This configuration achieved a 10.9× compression of the physical memory footprint and a 5.24× acceleration of inference throughput while maintaining high generalization capacity.

Finally, for high-performance cloud environments (GPU), the experiments demonstrated that native INT8 inference within the PyTorch framework remained inefficient without specialized inference optimization frameworks. While FP16 formatting remains the Pareto-optimal choice for massive linear architectures like VGG (achieving over 600 FPS), for inherently optimized residual networks like ResNet, standard 32-bit floating-point execution guided by Two-Stage Distillation (Method 06) provides the most efficient balance of speed and accuracy. These findings offer a clear roadmap for the deployment of specialized, hardware-aware AI solutions in real-time computer vision systems.

Despite the obtained results, future research will extend the proposed approach to modern lightweight architectures specifically designed for edge-oriented deployment scenarios. Furthermore, future studies will investigate fully optimized inference pipelines based on specialized vendor-specific frameworks and graph compilers, such as NVIDIA TensorRT, AMD MIGraphX, and Intel OpenVINO, to provide a more comprehensive assessment of INT8 quantization efficiency on dedicated hardware accelerators. Such extensions will enable a broader evaluation of hardware-aware compression strategies beyond the native framework execution environment considered in this study.

ACKNOWLEDGMENTS AND FUNDING SOURCES

The author(s) received no financial support for the research, writing, and/or publication of this article.

COMPLIANCE WITH ETHICAL STANDARDS

The authors declare that they have no conflicts of interest relevant to this study.

AUTHOR CONTRIBUTIONS

Conceptualization, [A.K., L.D.]; methodology, [A.K.]; validation, [A.K., L.D.]; formal analysis, [A.K.]; investigation, [A.K.]; resources, [A.K.]; data curation, [A.K.]; writing – original draft preparation, [A.K.]; writing – review and editing, [A.K., L.D.]; visualization, [A.K.]; supervision, [A.K., L.D.]; project administration, [A.K., L.D].

All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] Liu, Z., Mao, H., Wu, C. Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11976-11986). https://doi.org/10.1109/CVPR52688.2022.01167

[2] Deng, L., Li, G., Han, S., Shi, L., & Xie, Y. (2020). Model compression and hardware acceleration for neural networks: A comprehensive survey. In *Proceedings of the IEEE*, 108(4), 485-532. https://doi.org/10.1109/JPROC.2020.2976475

[3] Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6), 1789-1819. https://doi.org/10.1007/s11263-021-01453-z

[4] Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A. A., & Wilson, A. G. (2021). Does knowledge distillation really work? In *Advances in Neural Information Processing Systems (NeurIPS)* (Vol. 34, pp. 6906-6919). https://arxiv.org/abs/2106.05945

[5] Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., & Guttag, J. (2020). What is the state of neural network pruning?. *Proceedings of Machine Learning and Systems (MLSys)*, 2, 129-146. https://arxiv.org/abs/2003.03033

[6] Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., Van Baalen, M., & Blankevoort, T. (2021). A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*. https://arxiv.org/abs/2106.08295

[7] Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., & Keutzer, K. (2022). A survey of quantization methods for efficient neural network inference. In G. K. Thiruvathukal, Y.-H. Lu, J. Kim, Y. Chen, & B. Chen (Eds.), *Low-power computer vision: Improve the efficiency of artificial intelligence* (pp. 291–326). Chapman and Hall/CRC. https://doi.org/10.1201/9781003162810-13

[8] Ngo, T. T., Nguyen, B., Pham, T. B., Truong, S. N., Vo, H. M., Minh, K.-S., & Pham, K. V. (2026). Progressive compression of ResNet: An 8.33× reduction with 90.5% accuracy retention for edge deployment. *IEEE Access,* 14, 56900–56918. https://doi.org/10.1109/ACCESS.2026.3682854

[9] Lai, N., Dewi, D. A., Maidin, S. S., Xiao, W., Zhao, S., & Hu, Q. (2026). A comprehensive review of lightweight deep learning models for edge computing with future directions. *Discover Computing,* 29, 110. https://doi.org/10.1007/s10791-026-10021-3

[10] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), 1-74. https://doi.org/10.1186/s40537-021-00444-8

[11] Liebenwein, L., Baykal, C., Lang, H., Feldman, D., & Rus, D. (2020). Provable filter pruning for efficient neural networks. In *International Conference on Learning Representations*. https://arxiv.org/abs/1911.07412

[12] Wu, H., Judd, P., Zhang, X., Isaev, M., & Micikevicius, P. (2020). Integer quantization for deep learning inference: Principles and empirical evaluation. *arXiv preprint arXiv:2004.09602*. https://arxiv.org/abs/2004.09602

[13] Yun, J., Kang, B & Fu, Z. (2023). The hidden power of pure 16-bit floating-point neural networks. *arXiv preprint arXiv:2301.12809*. https://arxiv.org/abs/2301.12809

[14] Gyawali, D. (2023). Comparative analysis of CPU and GPU profiling for deep learning models. *arXiv preprint arXiv:2309.02521*. https://arxiv.org/abs/2309.02521

[15] Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., & Sun, J. (2021). RepVGG: Making VGG-style ConvNets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 13733-13742) https://doi.org/10.1109/CVPR46437.2021.01352

[16] Yao, Z., Dong, Z., Zheng, Z., Gholami, A., Yu, J., Tan, E., ... & Keutzer, K. (2021). HAWQV3: Dyadic neural network quantization. In *International Conference on Machine Learning* (pp. 11875-11886). https://arxiv.org/abs/2011.10680

АПАРАТНО-ОРІЄНТОВАНА ГІБРИДНА КОМПРЕСІЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ СИСТЕМ З ОБМЕЖЕНИМИ РЕСУРСАМИ

Андріан Козинець, Лідія Демків
Кафедра системного проектування,
Львівський національний університет імені Івана Франка,
вул. Драгоманова 50, м. Львів, 79005, УКРАЇНА

АНОТАЦІЯ

Вступ. Глибокі нейронні мережі стикаються з критичними обчислювальними бар'єрами для розгортання на периферійних пристроях з обмеженими ресурсами. Хоча існують окремі методи компресії, їх ізольоване застосування спричиняє значне зниження точності. Синергетичне поєднання обрізання (прунінгу), квантування та дистиляції знань пропонує рішення, проте їхня взаємодія та апаратно-залежна ефективність залишаються недостатньо дослідженими.

Матеріали та методи. Це дослідження розглядає гібридні конвеєри компресії, застосовані до архітектур згорткових нейронних мереж із послідовним з'єднанням шарів та архітектур із залишковими зв'язками. Оцінено алгоритм двоетапної дистиляції з додатковою попередньою фазою стабілізації класифікатора для усунення градієнтного шоку в поєднанні зі структурним видаленням фільтрів та статичним цілочисельним квантуванням. Кросплатформний аналіз проводився на центральних і графічних процесорах із побудовою меж ефективності Парето для визначення оптимальних компромісів.

Результати. Фаза попередньої стабілізації успішно усунула градієнтний шок, підвищивши точність моделей-учнів на величину до 2.6%. Ізольоване структурне обрізання значно погіршило точність, проте «лікування дистиляцією» за допомогою м'яких цілей учительської моделі дозволило обрізаній мережі (із видаленням майже 30% ваг) перевершити свою повнорозмірну базову лінію. На центральних процесорах (ЦП) гібридний конвеєр зменшив обсяг пам'яті в 11 разів і прискорив виведення у 5 разів. Натомість цілочисельне квантування виявилося неефективним на графічних процесорах (ГП) через затримки доступу до пам'яті. Для таких прискорювачів формат чисел із плаваючою комою половинної точності був Парето-оптимальним для масивних архітектур із послідовним з'єднанням шарів, тоді як формат чисел із плаваючою комою одинарної точності в поєднанні з двоетапною дистиляцією забезпечив найкращу ефективність для архітектур із залишковими зв'язками.

Висновки. Інтеграція двоетапної дистиляції знань зі структурним обрізанням та квантуванням принципово перевершує ізольовані методи компресії. Оптимальна стратегія суворо залежить від апаратного забезпечення та топології: цілочисельні гібридні конвеєри є необхідними для енергоефективних периферійних ЦП, тоді як на ГП вибір між половинною та стандартною точністю визначається структурними характеристиками конкретної мережі.

Ключові слова: Штучний інтелект, Глибоке навчання, Оптимізація нейронних мереж, Системи з обмеженими ресурсами, Апаратне прискорення.

Received / Одержано	Revised / Доопрацьовано	Accepted / Прийнято	Published / Опубліковано
10 June, 2026	08 July, 2026	10 July, 2026	28 September, 2026