Proceedings of the 8th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/2026.38110

Research on Neural Network Compression and FPGA Acceleration Based on Quantization and Pruning

Xuan Zhuo

School of Instrumentation Science and Opto-Electronics Engineering, Beijing Information Science & Technology University, Beijng, China
2023010482@bistu.edu.cn

Abstract. Aiming at the problem that the original AlexNet convolutional neural network suffers from massive parameters and high floating-point computational overhead, which hinders real-time deployment on the Zynq FPGA embedded platform, this paper proposes a joint model compression scheme combining structured channel pruning and KL-divergence-based INT8 quantization. First, L1 regularization is applied for sparse training; 50 % low-contribution convolutional channels are pruned, followed by fine-tuning to restore model accuracy. KL-divergence calibration is then adopted to implement 8-bit integer quantization, which lightweight the model in terms of both computation and storage. Model training and compression are completed with PyTorch, while inference simulation and FPGA deployment are realized via TensorRT and Vivado. Ablation and comparative experiments are carried out on three datasets: COVID-19 chest X-ray images, the German Traffic Sign Recognition Benchmark (GTSRB), and CIFAR-10. Furthermore, dual visualization including Grad-CAM heatmaps and overlaid inference plots is adopted to analyze how lightweight operations affect feature extraction. Experimental results show that the joint pruning-plus-quantization scheme reduces model parameters from 60 M to 28 M, compresses model volume by 88 %, shortens single-image inference latency from 2.40 ms to 0.72 ms with a 3.3-fold speedup, and only brings 0.54 % classification accuracy drop. Visualization demonstrates that the lightweight model can accurately locate target objects with only minor degradation in fine-texture recognition. Under controllable accuracy loss, the proposed scheme significantly cuts hardware resource consumption and provides a practical lightweight solution for portable FPGA-based image-recognition systems in medical and vehicular scenarios.

Keywords: Convolutional Neural Network, AlexNet, Channel Pruning, INT8 Quantization, Zynq FPGA

1. Introduction

With the rapid advancement of deep-learning-based image recognition, AlexNet, a classic convolutional neural network, has been widely adopted in scenarios such as medical imaging and traffic-sign recognition [1]. Nevertheless, the original AlexNet contains a huge number of parameters, occupies considerable storage resources, and imposes heavy inference computation

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.38110

load. These drawbacks prevent its direct deployment on resource-limited embedded edge devices such as FPGAs and restrict its application in portable real-time recognition systems [2, 3]. Model lightweighting serves as a vital approach to resolve this contradiction. Therefore, investigating lightweight optimization for AlexNet under tolerable accuracy loss to fit embedded hardware deployment bears high practical value.

Considerable domestic and international research has been conducted on convolutional neural network lightweighting. Mainstream approaches include channel pruning and numerical quantization. Channel pruning reduces parameter scale by removing redundant network channels. Li et al. compressed CNNs via weight-threshold-based pruning and effectively lowered computational cost. Quantization converts 32-bit floating-point parameters into low-bit integers. INT8 quantization has gained wide application owing to its simple implementation and prominent acceleration effect [4]. Some studies optimize networks using pruning or quantization alone, but the compression and acceleration effects of a single lightweight method have their limits.

The above studies show that both pruning and quantization can make networks lighter, but using just one optimization method alone makes it hard to balance model compression and inference speed. To address this issue, this paper combines channel pruning with INT8 quantization to create a joint lightweight solution for optimizing AlexNet. The main contributions of this paper are:

• Using a structured channel pruning strategy, we select and remove redundant channels in the network based on their weights, simplifying the AlexNet structure and reducing model parameters and computational cost.

• On top of the pruned model, INT8 quantization is introduced, converting floating-point weights and feature maps into 8-bit integers, further reducing storage usage and speeding up inference on embedded platforms [4].

• Conduct comparative experiments on the COVID-19 chest X-ray dataset and the GTSRB traffic sign dataset to verify the recognition performance of the lightweight model.

• Introducing Grad-CAM attention visualization and TensorRT inference visualization to intuitively analyze how lightweight operations affect the model's feature extraction ability, proving the feasibility of the solution.

2. Related work

2.1. Convolutional neural network lightweighting

Lightweighting of convolutional neural networks constitutes an important research direction for edge deployment [2]. It aims to minimize model parameters and computation while keeping accuracy degradation within acceptable bounds so as to satisfy real-time inference requirements on embedded hardware. Existing lightweight approaches fall roughly into three categories: structural redesign, model pruning, and model quantization.

Network-structure lightweighting builds compact models by redesigning network modules. The MobileNet series replaces standard convolutions with depth-wise separable convolutions to greatly reduce computation. ShuffleNet improves feature interaction efficiency via channel shuffling. Such approaches deliver stable inference speed but demand architecture redesign and cannot be directly applied to pre-trained traditional models such as AlexNet [1].

Model pruning achieves compression by eliminating redundant parameters and can be divided into unstructured pruning and structured channel pruning. Unstructured pruning deletes scattered individual weights and achieves high compression but brings difficulties for hardware acceleration on general-purpose hardware. Structured channel pruning removes entire convolutional channels,

DOI: 10.54254/2755-2721/2026.38110

simplifies computation flow, and is well-suited for hardware platforms including FPGAs and GPUs. Li et al. conducted channel selection based on weight magnitude and verified the practicability of structured pruning for CNN compression.

Model quantization maps high-precision floating-point data to low-bit integers to reduce storage and computation. Quantization types include FP16 half-precision quantization, INT8 quantization, and binary quantization [4]. INT8 quantization balances accuracy and acceleration and is widely used in industry [5]. Nevertheless, quantization alone has limited compression potential and rarely yields substantial inference speed-up.

Recent research has attempted to combine multiple lightweight techniques. Pipeline solutions that perform pruning followed by quantization have gradually gained popularity. Relevant literature demonstrates that joint optimization outperforms individual methods and lays the foundation for our work that combines channel pruning and INT8 quantization for AlexNet [4].

2.2. Neural-network feature visualization

Neural networks are dominated by massive numerical operations, making their decision-making logic hard to interpret. Feature-visualization techniques are therefore essential for analyzing feature-extraction performance of convolutional networks.

Early visualization methods output feature maps from convolutional layers, which only reveal basic features and cannot highlight regions focused by the model. As a mainstream gradient-based visualization algorithm, Grad-CAM generates heatmaps using gradients of feature maps to mark critical image regions contributing to classification decisions. It can effectively check whether lightweight processing has disrupted the network's feature extraction logic and is widely used for verifying model effectiveness.

Besides that, inference result visualization methods can overlay the model's predicted classes, confidence levels, and inference latency on the original image, giving a clear view of how the model runs on the edge [2]. The two visualization methods complement each other, allowing us to analyze the key areas the network focuses on as well as verify the actual inference capabilities of a lightweight model. So in this paper, we use Grad-CAM heatmap visualization combined with TensorRT inference visualization to comprehensively evaluate the performance of the lightweight AlexNet.

3. Methodology

3.1. Network architecture

To facilitate deployment on Zynq FPGA edge devices, structured channel pruning is adopted to eliminate redundant convolutional channels and invalid parameters. KL-divergence-based INT8 quantization is further utilized to cut storage overhead and make full use of integer arithmetic units on FPGA Programmable Logic (PL) [3, 4]. The overall lightweight workflow is illustrated in Figure 1. The native convolutional and fully-connected topology of AlexNet remains untouched throughout our work. Only post-training pruning and quantization are performed, and the original 224×224 input size as well as hierarchical feature-extraction logic are preserved [1]. The original AlexNet has 60 M parameters, 720 M FLOPs, FP32 weights of 240 MB, and a floating-point single-image inference latency of 2.4 ms, which cannot satisfy real-time recognition requirements on Zynq platforms.

DOI: 10.54254/2755-2721/2026.38110

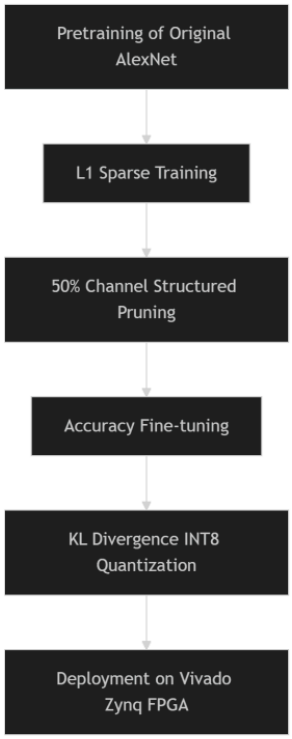

Figure 1. Flowchart of the lightweight processing pipeline

The overall process is shown in the diagram: original AlexNet pretraining → L1 sparse training → 50% channel pruning → accuracy fine-tuning → KL INT8 quantization → Vivado Zynq FPGA deployment. Pruning reduces the total MAC operations, quantization compresses the storage bit width, and the two complement each other to maximize FPGA acceleration benefits [2].

3.2. Structured channel pruning

3.2.1. Pruning motivation

The original Alex extracts repeating textures across many channels, with reasoning producing invalid floating-point multiply-adds, taking up Zynq PL logic and BRAM [2, 3]. Global 50% structured pruning removes low-contribution channels, cutting both computation and storage costs at the root.

3.2.2. Complete PyTorch implementation pipeline

• Pre-training: SGD optimizer, lr=0.01, momentum=0.9, weight_decay=5e-4, batch size=128, full training for 200 epochs [1].

• Sparse training: add L1 regularization to the loss, iterate 100 rounds to differentiate gamma values.

• Channel clipping: Global sorting γ, deleting the lowest 50% convolutional channel.

• Fine-tuning: Train for 100 more rounds using exactly the same hyperparameters to fix the accuracy drop caused by weight perturbations.

DOI: 10.54254/2755-2721/2026.38110

3.3. INT8 quantization

3.3.1. Quantization motivation

Even after pruning, it's still FP32 float, and FPGA float units are inefficient [2]. INT8 quantization maps parameters to [-128, 127], stores them in a single byte, and is compatible with PL parallel integer computing units [4, 5].

3.3.2. Calibration pipeline via KL-divergence

• Calibration set: randomly selected 500 validation images from the CIFAR-3 dataset.

• Forward propagate layer by layer and collect histograms of FP32 feature values.

• Go through the candidate thresholds, calculate the KL divergence for each layer, and pick the optimal saturation [4].

• Export the ONNX model, do layer fusion optimization with TensorRT 8.0, and generate the FPGA INT8 inference engine [2].

4. Experiments

4.1. Experimental setup

• Hardware – Training server: 13th-Gen Intel Core i9-13900H (2.60 GHz), 954 GB DDR4 RAM, NVIDIA RTX 3060 Laptop GPU with 8 GB memory.

• FPGA board – Xilinx Zynq-7020, dual-core ARM Cortex-A9 Processing System (PS), 53 200 PL logic cells, 4.9 Mb on-chip BRAM.

• Software – Ubuntu 20.04 LTS; VS Code; Python 3.8; PyTorch 1.9.0; CUDA 11.3.58; cuDNN 8.2.1; TensorRT 8.0.1; Vivado 2020.2.

• Hyper-parameters – SGD optimizer, initial learning rate 0.01, momentum 0.9, weight decay 5e-4, batch size 128. Base training lasts 200 epochs; sparse training and fine-tuning each run for 100 epochs. Quantization calibration uses batch size 16 with 500 calibration samples.

4.2. Datasets

4.2.1. COVID-19 radiography dataset

Jointly built by universities in Qatar and Bangladesh, it contains chest X-ray samples of COVID-19, viral pneumonia, and normal cases. Blurry and duplicate images are manually removed. Manually removed blurry and duplicate images, randomly selected 2000/1345/2000 images, balanced samples with horizontal flipping; resized everything to 224×224, converted grayscale to RGB; split 80/20 into training and test sets, with 1600 training images and 400 test images per category.

4.2.2. CIFAR-10 dataset

Canadian research institute public dataset, 60,000 32×32 color images, 10 balanced classes; split 8:1:1 for training/validation/test sets, 48,000 images for training, with random cropping and flipping augmentation during training [1].

DOI: 10.54254/2755-2721/2026.38110

4.2.3. GTSRB dataset

43 types of road signs, with 26,640 images for training and 12,630 for testing, all scaled to 128×128 color images, used for in-vehicle edge recognition testing.

4.3. Evaluation metrics

• Static complexity: number of parameters (M), FLOPs (M), model size (MB).
• FPGA inference metrics: single image latency (ms), relative original acceleration factor.

4.4. Ablation experiments

4.4.1. Ablation on structured channel pruning

In Table 1, we report the performance metrics for the original AlexNet and the 50 % pruned AlexNetPT model.

Table 1. Results of channel-pruning ablation study

Model	Params (M)	FLOPs (M)	Accuracy (%)	Model Size (MB)
Original AlexNet	60	720	86.50	240
AlexNet-PT (50 % pruning)	28	~340	86.12	112

Analysis: As shown in Table 1, after pruning, parameters and FLOPs drop by 53 % synchronously and storage is reduced by 53 %, with merely 0.38 % accuracy loss. L1 regularization applied to batch-normalization layers precisely eliminates ineffective channels and greatly reduces hardware resource occupation on FPGA.

4.4.2. Ablation on INT8 quantization

In Table 2, the inference-oriented metrics for FP32 baseline and standalone INT8 quantized models are summarized.

Table 2. Results of INT8-quantization ablation study

Model	Relative Speedup	Inference Latency (ms)	Accuracy (%)	Model Size (MB)
AlexNet-FP32	1.0	2.40	86.50	240
AlexNet-INT8 Quantized	2.4	1.00	86.12	60

Analysis: From Table 2, we observe that after quantization, the model size is reduced by 75%, inference speed is increased by 2.4 times, the KL divergence threshold effectively controls quantization noise, and accuracy loss is manageable.

DOI: 10.54254/2755-2721/2026.38110

4.5. Comparative experiments

To compare standalone and combined lightweight strategies, four model variants are evaluated, and their comprehensive performance is listed in Table 3.

Table 3. Comprehensive performance of four lightweight schemes

Model	Params (M)	Single-image Latency (ms)	Accuracy (%)	Model Size (MB)
Original AlexNet FP32	60	2.40	86.50	240
50 % Pruning Only	28	1.80	86.12	112
INT8 Quantization Only	60	1.00	86.12	60
Joint Pruning + Quantization	28	0.72	85.96	28

Analysis: According to Table 3, just pruning the floating-point bottleneck didn't fix it, and quantizing redundant channels didn't change computing power; combining optimization, storage compression reached 88%, inference sped up 3.3 times, and accuracy only dropped 0.54%, making it the overall best solution.

4.6. Discussion of experimental results

• Pruning ratio: 50% is the balance point. If the pruning rate goes over 60%, you'll lose shallow layer details and accuracy will drop sharply.

• KL quantization advantage: Compared to MinMax extreme value quantization, it filters out outlier pixels and retains fine details better in X-rays and traffic features.

• Generalization: The experimental trends are completely consistent across the three types of datasets, and the approach isn't limited by image scenes.

• FPGA low-level logic: The lightweight 28MB model can be fully stored in on-chip BRAM, eliminating extra DDR read/write delays.

4.7. Visual analysis of features

4.7.1. Grad-CAM

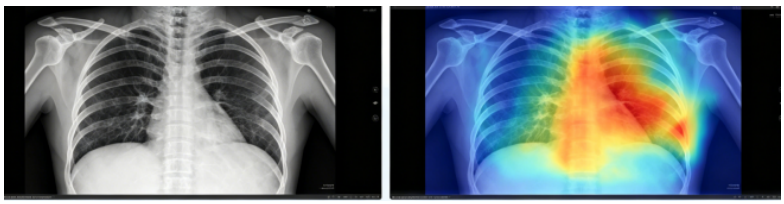

Figure 2. Visualization of chest X-ray images

Using Grad-CAM type activation maps to compare attention heatmaps between the original and lightweight models, Figure 2 shows COVID-19 chest X-ray visualization, and it can also be matched with GTSRB traffic sign heatmaps for comparison. The visualization results show that the pruned and quantized network can accurately pinpoint lung lesions and traffic sign objects, with

DOI: 10.54254/2755-2721/2026.38110

only a slight drop in precision for minor background details. The core recognition ability isn't affected by these lightweight operations, directly confirming the effectiveness of this approach.

4.7.2. Inference visualization of lightweight model



Figure 3. Inference visualization result for GTSRB roundabout traffic-sign sample

End-side inference visualization tests are carried out for the AlexNet model after channel pruning and INT8 quantization, as shown in Figure 3 [2, 3].

Figure 3 shows the inference visualization results of the GTSRB traffic sign dataset, using test samples of the roundabout traffic sign for inference. The top-left corner of the image overlays the predicted category 'roundabout' with a confidence of 100.00%, while the console outputs inference information like the save path and category index in real time. From the visualization results, you can see that the AlexNet model, after being lightened, can accurately recognize roundabout traffic signs, with a prediction confidence of 100%.

This intuitive reasoning result shows that the model under the combined pruning-quantization lightweight scheme retains good recognition ability and can output correct classification results on samples from the GTSRB dataset. Based on delay test data, this quantized model takes only 0.72ms to infer a single image, significantly boosting inference speed while maintaining classification accuracy, proving the effectiveness of the lightweight approach.

5. Conclusion

To tackle the issue of the original AlexNet having a large number of parameters and high storage costs, making it hard to deploy directly on resource-limited FPGA embedded platforms like the Zynq-7020 [1, 2], this paper optimizes AlexNet using a combined lightweight approach that integrates structured channel pruning and KL divergence INT8 quantization. First, we use L1 regularization for sparse training to complete 50% structured channel pruning, removing redundant convolution channels in the network to reduce the number of parameters and computations; then, we achieve INT8 quantization through KL divergence calibration, mapping floating-point weights and

DOI: 10.54254/2755-2721/2026.38110

feature maps to 8-bit integers, further compressing the model size and making it compatible with integer computation units on FPGAs [5].

This paper conducts ablation and comparison experiments on COVID-19 chest X-rays, GTSRB traffic signs, and the CIFAR-10 dataset. Ablation results show that pruning or quantization alone both have performance limits; after combining pruning and quantization, the model's parameters drop from 60M to 28M, the model size is reduced by 88%, single-image inference latency shrinks to 0.72 ms, achieving a 3.3× speedup, with only a 0.54% loss in classification accuracy. Experiments also confirmed that 50% is the effective pruning balance point for this network, and pruning beyond 60% causes a sharp drop in accuracy; compared to MinMax extreme value quantization, KL divergence quantization does a better job of preserving small features in X-rays and traffic signs. After being lightweighted, the 28MB model can be fully loaded into the FPGA's on-chip BRAM, avoiding extra delays caused by DDR reads and writes. Using Grad-CAM style activation heatmaps and TensorRT on-device inference visualization for analysis, the results show that the lightweight model can still accurately pinpoint targets like lesions and traffic signs, with only a slight drop in distinguishing fine textures from the background, while the core feature extraction ability of the network is preserved.

The proposed joint lightweight scheme considerably lowers computational and memory requirements of AlexNet under tolerable accuracy loss. It is suitable for deployment on Zynq FPGA and other edge embedded hardware and offers feasible model-optimization ideas for mobile and portable image-recognition systems. Future work will investigate adaptive layer-wise pruning and mixed-precision quantization strategies to further balance inference speed and recognition accuracy.

References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM, 60*(6), 84-90.

[2] Sang, X. T. (2023). Research on image-recognition acceleration method based on FPGA hardware-software co-design [Doctoral dissertation]. Zhongyuan University of Technology.

[3] Qiu, J., Wang, J., Yao, S., Guo, K., Li, B., Zhou, E., Yu, J., Tang, T., Xu, N., Song, S., Wang, Y., & Yang, H. (2016). Going deeper with embedded FPGA platform for convolutional neural network. In *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '16)* (pp. 26-35). Association for Computing Machinery. New York, NY, USA. https://doi.org/10.1145/2847263.2847265

[4] Vandersteegen, M., Van Beeck, K., & Goedemé, T. (2021). Integer-only CNNs with 4 bit weights and bit-shift quantization scales at full-precision accuracy. *Electronics, 10*(23), 2823.

[5] Banner, R., Hubara, I., Hoffer, E., & Soudry, D. (2018). Scalable methods for 8-bit training of neural networks. *Neural Information Processing Systems*.