Adaptive Quantization and Pruning of Deep Neural Networks via Layer Importance Estimation

NEURAL INFORMATION PROCESSING SYSTEMS

IIT MADRAS ZANZIBAR

Tushar Shinde

School of Engineering and Science, Indian Institute of Technology Madras Zanzibar, Tanzania

INTRODUCTION

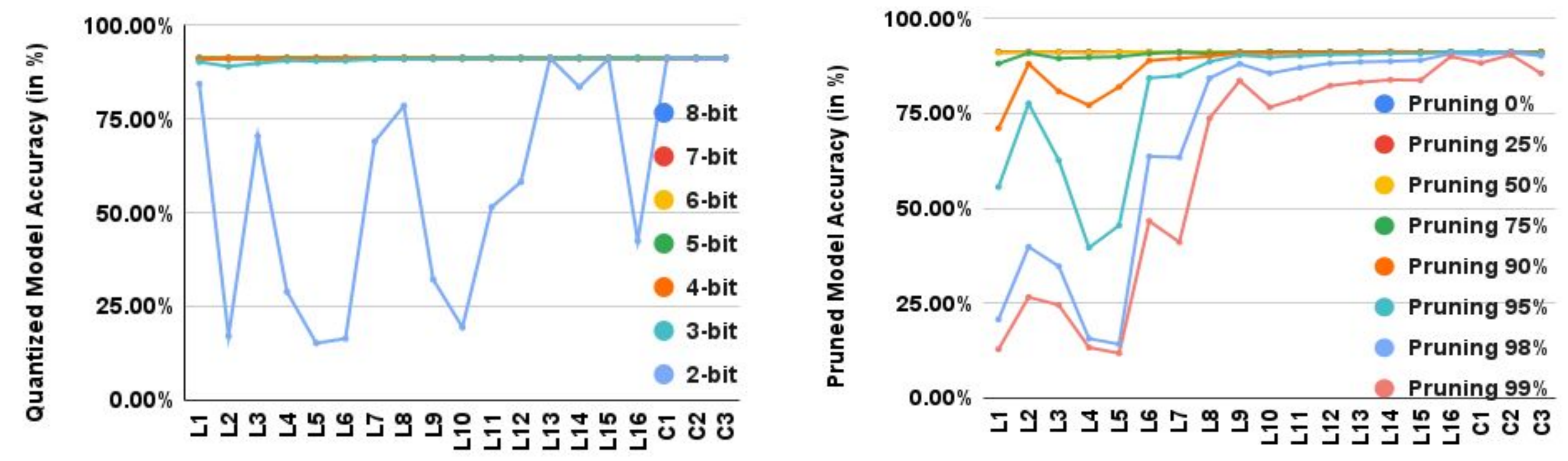
- Deep Neural Networks (DNNs) achieve state-of-the-art performance in domains like computer vision and speech processing.
- **Deployment Challenges:** High computational and storage demands make DNNs unsuitable for resource-constrained edge devices.
- Traditional uniform quantization and pruning often fail to maintain accuracy, as layers contribute unequally to model performance.

OBJECTIVES

- Introduce a **layer-wise quantization** method that assigns bit-widths based on layer importance to optimize model size without compromising accuracy.
- Design an **adaptive pruning** strategy that identifies and prunes less important parameters effectively while maintaining model performance.
- **Combine Quantization and Pruning:** Investigate the synergy of adaptive quantization and pruning to achieve compact yet accurate DNNs.

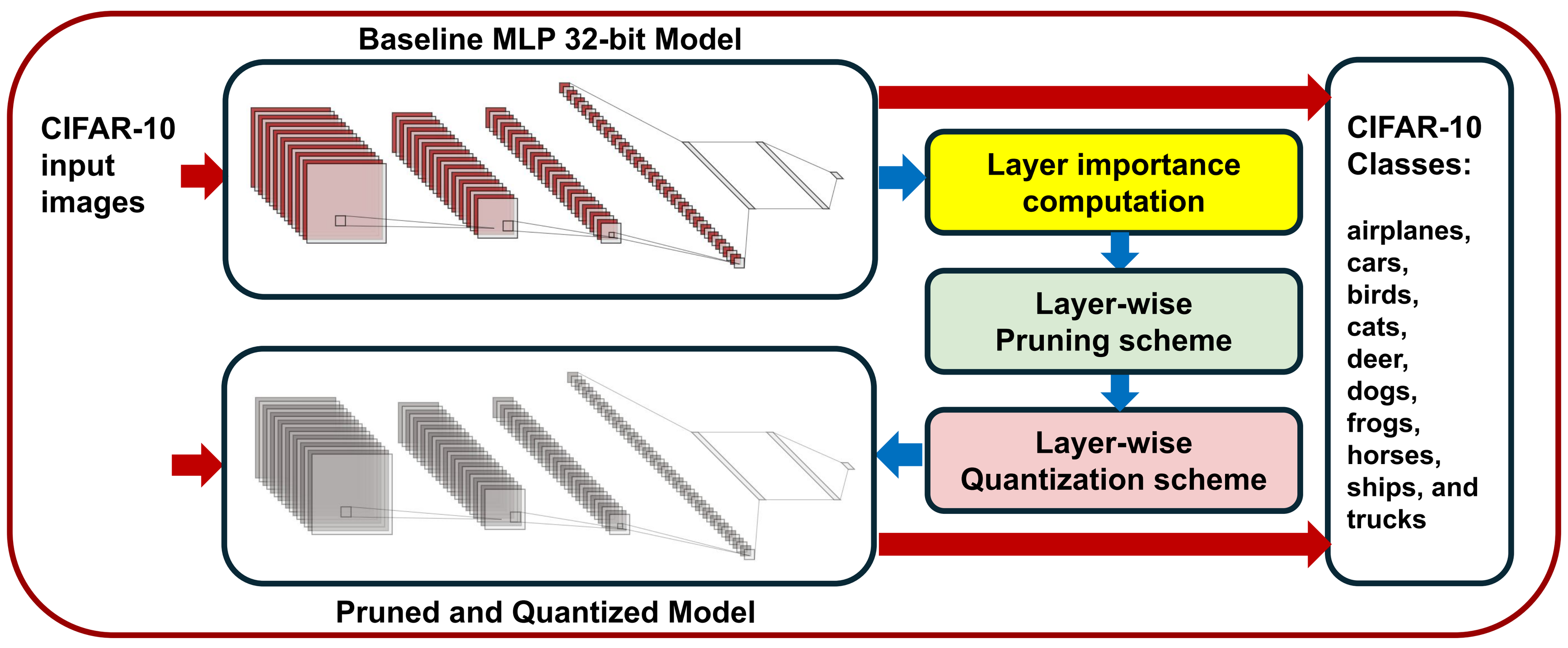
PROBLEM ANALYSIS & MOTIVATION

- **Motivation:** Fixed compression approaches often overlook the varying importance of different layers in DNN, leading to performance degradation.
- Mixed-precision approaches could improve efficiency.
- **Solution:** Adaptive quantization and pruning methods that tailor bit-widths and sparsity thresholds layer-wise for optimal trade-offs.



Analysis of individual layers: a) varying bit-precisions and b) varying pruning percentages

PROPOSED METHOD



- **Layer Importance** computation using metrics like normalized parameter proportion, layer entropy, layer variance, and layer sparsity.

$$N_P(l) = \frac{\text{Parameters in layer } l}{\text{Total parameters in the model}} \qquad N_V(l) = \log\left(e - 1 + \frac{\text{Variance of layer } l}{\max_k(\text{Variance of layer } k)}\right)$$

$$N_E(l) = \frac{\text{Entropy of layer } l}{\text{Bit-precision of the model}} \qquad S(l) = \frac{\text{Number of zero or near-zero activations in layer } l}{\text{Total number of activations in layer } l}$$

$$\text{Importance}(l) = w_P \cdot N_P(l) + w_E \cdot N_E(l) + w_V \cdot N_V(l) + w_S \cdot S(l)$$

- **CIFAR-10 classification task** is a standard problem in ML and computer vision, where the goal is to classify images into one of 10 categories.
- **Layer Importance** is computed to guide quantization and pruning decisions.
- **Iterative Optimization:** Layers ranked by importance. Sequential optimization adjusts bit-width and pruning thresholds, validating performance at each step.
- **Layer-wise Adaptive Pruning:** Adapts pruning per layer to balance size reduction and accuracy. Iterative optimization ensures maximal pruning with tolerable accuracy loss. Adaptive pruning threshold is tuned for each layer.

$$\hat{W}_{l,ij} = \begin{cases} W_{l,ij} & \text{if } |W_{l,ij}| > Z_T(l) \\ 0 & \text{if } |W_{l,ij}| \leq Z_T(l) \end{cases} \qquad Z_T(l) = k(l) \times \sigma(l)$$

- **Layer-wise Adaptive Quantization:** Adapts bit-width per layer to balance size reduction and accuracy. Iterative optimization ensures minimal bit-width with tolerable accuracy loss.
- **Performance Threshold:** Ensures that performance degradation remains well within an acceptable margin, ensuring model reliability during model pruning and quantization.

$$T_{\text{margin}}(l) = T_{\text{margin}} \times \text{Importance}(l)$$

RESULTS

- **Dataset:** CIFAR-10 dataset comprises 60,000 32x32 color images across 10 classes, with data normalized to [0, 1] and augmented via horizontal flips and random crops.
- **DNN Models Tested:** VGG19, ResNet18, ResNet34
- **Implementation details:**
  - Trained from scratch for 100 epochs.
  - SGD optimizer with learning rate=0.02, batch size=128, and Cross-Entropy loss.
- **Hyper-parameters Setting**:
  - All weights are equal (sum to 1) for layer importance computation.
  - Weight quantization from 1-bit to 8-bit precision.
  - Pruning thresholds started from 99.7% (k(l)=3) to no pruning (k(l)=0).
- **Evaluation Metrics:** Accuracy & average bit-width $\bar{b} = \sum_{l=1}^{L} b(l) \cdot S(l) \cdot N_P(l)$

- **Baseline:** VGG19: 91.16%, ResNet18: 86.06%, ResNet34: 86.22%.
- **Fixed-bit quantization** (e.g., 3-bit) led to significant accuracy drops.
- **Fixed pruning** (e.g., 75%) too resulted in notable accuracy degradation.
- **Our adaptive method** (Quantization only, Pruning only, and Combined) achieves near-baseline accuracy while significantly reducing model size.
- **Model Comparison:** Proposed Method vs. APoT and LIEI-NNQ:
  - Existing methods suffered higher accuracy losses at low average bit-widths.
  - Our approach maintained accuracy at significantly lower average bit-widths.
- **Average bit-width reduction:** Quantization (combined pruning+Q)
  - VGG19: 2.24 bits (1.08),
  - ResNet18: 3.41 bits (2.66),
  - ResNet34: 4.18 bits (2.42).

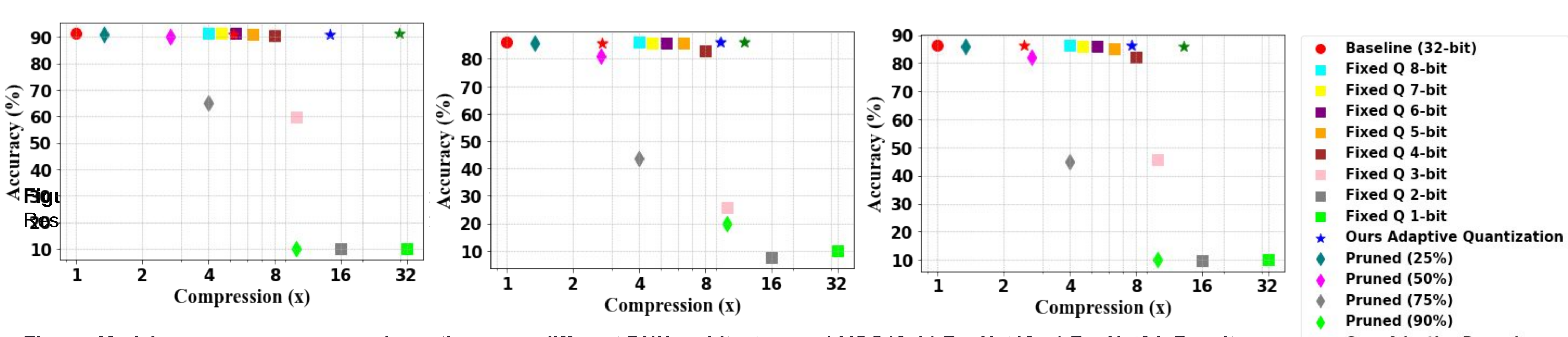


Figure: Model accuracy vs. compression ratio across different DNN architectures: a) VGG19, b) ResNet18, c) ResNet34. Results for different variants of fixed and adaptive pruning and quantization approaches. Our method is highlighted with star marker.

Table: Performance comparison with existing methods across DNNs

Method	Model	#Parameters (M)	Avg. bit-width	Parameters Size (MB)	Accuracy difference (in %)
Proposed AQP	VGG19	20.04	1.08	2.72	**0.00%**
Proposed AQP	ResNet18	11.69	2.66	3.17	0.00%
Proposed AQP	ResNet34	21.8	2.42	6.52	-0.09%
APoT	ResNet18	11.69	4	5.87	-0.40%
APoT	ResNet18	11.69	3	4.38	-0.84%
APoT	ResNet18	11.69	2	2.92	-1.75%
LIEI-NNQ	ResNet18	11.69	1.96	2.77	-1.55%
APoT	MobileNetV2	3.47	4	1.74	-4.25%
APoT	MobileNetV2	3.47	3	1.30	-10.39%
APoT	MobileNetV2	3.47	2	0.87	-24.45%
LIEI-NNQ	MobileNetV2	3.47	3.32	1.45	-9.42%

CONCLUSION

- **Contribution:** Introduced an adaptive layer-wise quantization and pruning method for enhancing DNN efficiency while preserving accuracy.
- **Results:** The adaptive approach maintained accuracy with minimal loss across varying bit-widths. Outperformed uniform quantization and pruning techniques..
- **Architecture Advantage:** Our approach optimizes each layer's precision, achieving efficient models, ideal for resource-constrained devices.
- **Limitations:** Our method may be influenced by the weight values used in layer importance computation, requiring further investigation.
- **Future Work:** Investigate other advanced model compression techniques for further model optimization on diverse datasets to validate its generalizability.

REFERENCES

[1] Krizhevsky, A. and Hinton, G., 2009. Learning multiple layers of features from tiny images.
[2] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
[3] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[4] Liu, H., Elkerdawy, S., Ray, N. and Elhoushi, M., 2021. Layer importance estimation with imprinting for neural network quantization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2408-2417).
[5] Li, Y., Dong, X. and Wang, W., 2019. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. arXiv preprint arXiv:1909.13144

ACKNOWLEDGMENTS

This work was in part supported by the Walmart Center of Technical Excellence (IIT Madras)} Project Grant Award.

Website Paper